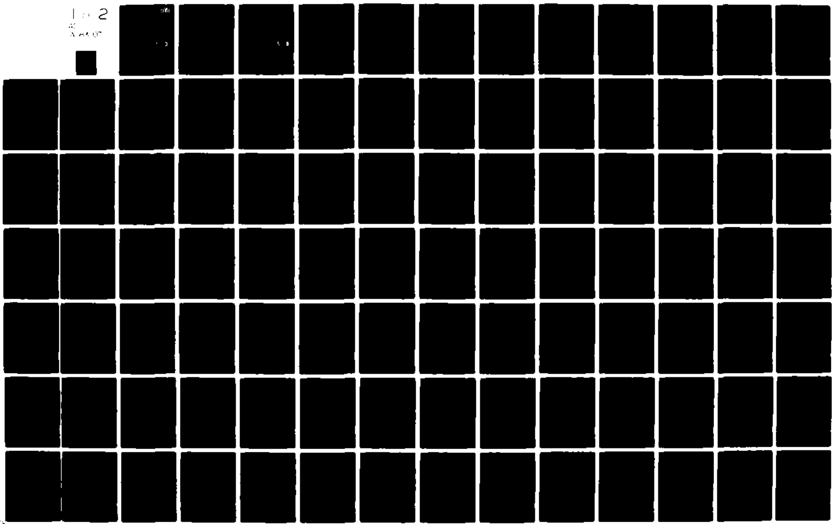
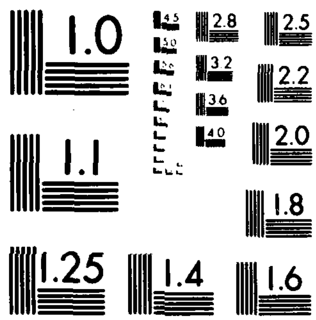
AD-A083 207 BOEING AEROSPACE CO SEATTLE WA F/G 9/2
REQUIREMENTS SPECIFICATION. ONE OF THE SOFTWARE ACQUISITION ENG--ETC(U)
JAN 79 D C WHITMORE F33657-76-C-0723
UNCLASSIFIED D180-20676-1 ASD-TR-78-45 NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963 A

ASD-TR-78-45

(2) LEVEL II

AD A083207

REQUIREMENTS SPECIFICATION
One of the Software Acquisition Engineering Guidebook Series

DIRECTORATE OF EQUIPMENT ENGINEERING
DEPUTY FOR ENGINEERING

JANUARY 1979

TECHNICAL REPORT ASD-TR-78-45
Final Report

Approved for public release; distribution unlimited.

AERONAUTICAL SYSTEMS DIVISION
AIR FORCE SYSTEMS COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OHIO 45433

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

This report has been reviewed by the Information Office (OI) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations.

This technical report has been reviewed and is approved for publication.

Richard W. Ittelson

RICHARD W. ITTELSON,
Technical Advisor
Directorate of Equipment Engineering

Richard J. Sylvester

RICHARD J. SYLVESTER,
ASD Weapon Systems Computer Resource
Focal Point
Deputy for Engineering

FOR THE COMMANDER

John S. Kubin

JOHN S. KUBIN, Colonel, USAF
Director, Equipment Engineering

"If your address has changed, if you wish to be removed from our mailing list, or if the addressee is no longer employed by your organization please notify ____________,W-PAFB, OH 45433 to help us maintain a current mailing list".

Copies of this report should not be returned unless return is required by security considerations, contractual obligations, or notice on a specific document.

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

19 REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER 18 ASD-TR-78-45

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER 9

4. TITLE (and Subtitle) 6 REQUIREMENTS SPECIFICATION, 9. One of the Software Acquisition Engineering Guidebook Series.

5. TYPE OF REPORT & PERIOD COVERED Final rept.

6. PERFORMING ORG. REPORT NUMBER 14 D180-20676-1

7. AUTHOR(s) 10 D.C. Whitmore

8. CONTRACT OR GRANT NUMBER(s) 15 F33657-76-C-0723

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Boeing Aerospace Company
P.O. Box 3999
Seattle, Washington 98124

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
PE64740F 16
Project 2238

11. CONTROLLING OFFICE NAME AND ADDRESS
HQ. ASD/ENE
Wright-Patterson AFB, OH 45433

12. REPORT DATE 11 2 Jan [redacted] 79

13. NUMBER OF PAGES 104

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) 12 100

15. SECURITY CLASS. (of this report) Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for Public Release, Distribution Unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Software Acquisition, Acquisition Engineering, Requirements Specification, Software Requirements, Top Down Structured Programs, Source Selection, Program Design Language, Test Requirements Document, Software Specifications

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report is one of a series of guidebooks whose purpose is to assist Air Force Program Office Personnel and other USAF acquisition engineers in the acquisition engineering of software for Automatic Test Equipment and Training Simulators. This guidebook describes the requirements specification process. Acquisition engineering tasks are described starting with definition of using command needs and continuing through procurement contract.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

059610 slt

FOREWORD

This guidebook was prepared as part of the Software Acquisition Engineering Guidebooks contract, F33657-76-C-0723. It describes the process for deriving requirement specifications for ground systems; i.e., training simulators and automatic test equipment. Acquisition engineering tasks are defined and described for specification evolution from initial analysis of user needs through final negotiation of the procurement contract.

This guidebook is one of a series intended to assist the Air Force Program Office and engineering personnel in software acquisition engineering for automatic test equipment and training simulators. Titles of other guidebooks in the series are listed in the introduction. These guidebooks will be revised periodically to reflect changes in software acquisition policies and feedback from users.

This guidebook reflects an interpretation of DOD directives, regulations and specifications which were current at the time of guidebook authorship. Since subsequent changes to the command media may invalidate such interpretations, the reader should also consult applicable government documents representing authorized software acquisition engineering processes.

This guidebook contains alternate recommendations concerning methods for cost-effective software acquisition. The intent is that the reader determine the degree of applicability of any alternative based on specific requirements of the software acquisition with which he is concerned. Hence, the guidebook should only be implemented as advisory rather than as mandatory or directive in nature.

This Software Acquisition Engineering Guidebook is one of a series prepared for Aeronautical Systems Division, Air Force Systems Command, Wright-Patterson AFB OH 45433. Inquiries regarding guidebook content should be sent to ASD/ENE, Wright-Patterson AFB OH 45433. The following list presents the technical report numbers and titles of the entire Software Acquisition Engineering Guidebook Series. Additional copies of this guidebook or any other in the series may be ordered from the Defense Documentation Center, Cameron Station, Alexandria VA 22314.

ASD-TR-78-43,	Computer Program Maintenance
ASD-TR-78-44,	Software Cost Measuring and Reporting
ASD-TR-78-45,	Requirements Specification
ASD-TR-78-46,	Computer Program Documentation Requirements
ASD-TR-78-47,	Software Quality Assurance
ASD-TR-78-48,	Software Configuration Management
ASD-TR-78-49,	Measuring and Reporting Software Status
ASD-TR-78-50,	Contracting for Software Acquisition
ASD-TR-79-5042,	Statements of Work (SOW) and Requests for Proposal (RFP)
ASD-TR-79-5043,	Reviews and Audits
ASD-TR-79-5044,	Verification, Validation and Certification
ASD-TR-79-5045,	Microprocessors and Firmware
ASD-TR-79-5046,	Software Development and Maintenance Facilities
ASD-TR-79-5047,	Software Systems Engineering
ASD-TR-79-5048,	Software Engineering (SAE) Guidebooks Application and Use

TABLE OF CONTENTS

TABLE OF CONTENTS - Continued

LIST OF FIGURES

LIST OF TABLES

Section 1.0 INTRODUCTION

The availability and performance of modern weapon systems, including ground support systems, depend critically on the subsystems which operate under the control of software. Ground system performance, in turn, hinges on how well the functional and design requirements for hardware and software have been specified. These requirements are the result of a derivation process encompassing the discipline of both weapon system engineering and computational system engineering. This process of requirements derivation - especially software requirements - is the principal topic of this guidebook. It is described in terms of analyses and studies that are performed and how these relate to system development phasing. The particular systems with which this guidebook is concerned are automatic test equipment and training simulators.

1.1 PURPOSE

The primary purpose of this guidebook is to assist AF engineering personnel directly responsible for Training Simulators (TS) and Automatic Test Equipment (ATE) software acquisition to assure the performance requirements for this software are successfully monitored and developed. The guidebook should also be helpful to Air Force managers responsible for the procurement of the total TS or ATE systems.

1.2 SCOPE

This is one of a series of guidebooks related to the Software Acquisition Engineering (SAE) process for TS and ATE ground-based systems. Other SAE guidebook titles are listed below:

Software Cost Measuring and Reporting

Requirements Specification

Contracting for Software Acquisition Statement of Work (SOW) and Requests for Proposal (RFP)

Regulations, Specification and Standards

Measuring and Reporting Software Status

Computer Program Documentation Requirements

Software Quality Assurance

Verification

Validation and Certification

Computer Program Maintenance

Software Configuration Management

Reviews and Audits

Management Reporting

For the purposes of this guidebook, TS requirements specification may be defined as the process which starts with the gleaning of requirements from a basic statement of need, such as in a Required Operational Capabilities (ROC) document issued by a using AF echelon and ends with the collection of the approved requirements in a development (Part I) specification. This process is managed by the Air Force but involves participation by other organizations; e.g., weapon system prime contractor and ground system suppliers.

For ATE, the process normally starts with an involved set of maintenance and repair analyses performed by the mission system prime contractor. This identifies the support equipment which will be required for mission support and also differentiates between normal and automatic

equipment. The ATE development (Part I) specifications are written by the mission system prime contractor who can either develop or procure the ATE.

1.3 TS AND ATE OVERVIEW

The purpose of this section is to provide a brief sketch of TS and ATE system characteristics, including the function of the software associated with each.

1.3.1 TS System Characteristics

The TS system is a combination of specialized hardware, computing equipment, and software designed to provide a synthetic flight and/or tactics environment in which aircrews learn, develop and improve the techniques associated with their individual tasks in a specific type aircraft. In many cases, visual, aural, and/or motion systems may be included. Figure 1.3-1 depicts a typical training simulator which employs digital processing capability.

The computer system, integral to the crew training simulator, consists of one or more general purpose computers. The computing hardware consists of machines with hardware floating point arithmetic and sufficient bit capacity to provide efficient use of the simulator High Order Language (HOL).

When a multi-processor/multi-computer system is used, it must be designed such that all computers operate in parallel in real-time and are controlled and time synchronized from a single computer program supervisor/executive. The executive directs the program execution and established priorities.

The simulator accepts control inputs from the trainee via cockpit controls (or other crew station controls) or from the instructor operator station, performs a real-time solution of the simulator mathematical model, and provides outputs necessary to accurately represent the static and dynamic behavior of the real world system within specified tolerance and performance criteria. Since training simulators are a combination of interdependent hardware and software, a joint development effort is required. As the complexity of training simulators increases, simulation software continues to grow in complexity, size, and cost. Software costs can and do exceed computer hardware costs in many cases. Therefore, it is imperative that the simulation software acquisition engineering process be subjected to formal system engineering planning and discipline to ensure effective and efficient simulator procurement.

1.3.2 ATE System Characteristics

ATE is defined as that equipment which is used for maintenance activities - principally in support of large deployed systems. ATE is used in place of manual devices either because it is more cost effective or the item being tested requires the speed and timing which only an automatic tester can achieve.

Figure 1.3-2 shows the typical components of an ATE system. Note that there are both hardware and software elements involved. Most of the elements shown will be found in one form or another in the majority of ATE systems.

The controls and displays section consists of the computer and peripheral devices like control panels, magnetic tape cassettes or disks, a cathode ray tube (CRT) and keyboard, and usually a small printer. The computer, as controlled by software, performs tasks like operating the peripheral devices, switching test stimuli on and off, and measuring and comparing responses of the Unit Under Test (UUT) to predetermined values. The operator will maintain ultimate control of the testing process through some of the peripherals. However, his interaction is usually minimal since, by definition, the automatic test feature was selected in preference to an operator-controlled test system. It is normally designed to allow a single configuration of ATE to be used for testing several articles of system equipment.

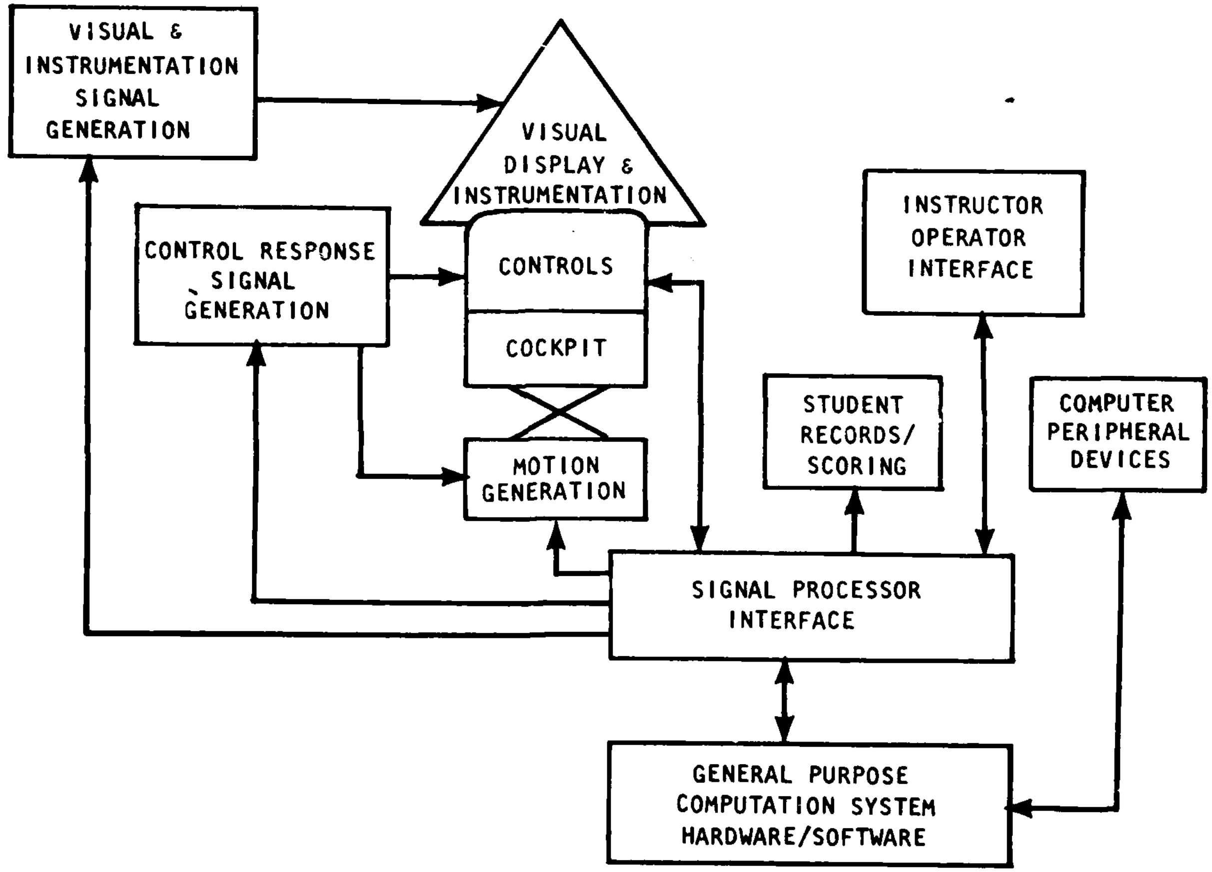


Figure 1.3.1. Typical Crew Training Simulator

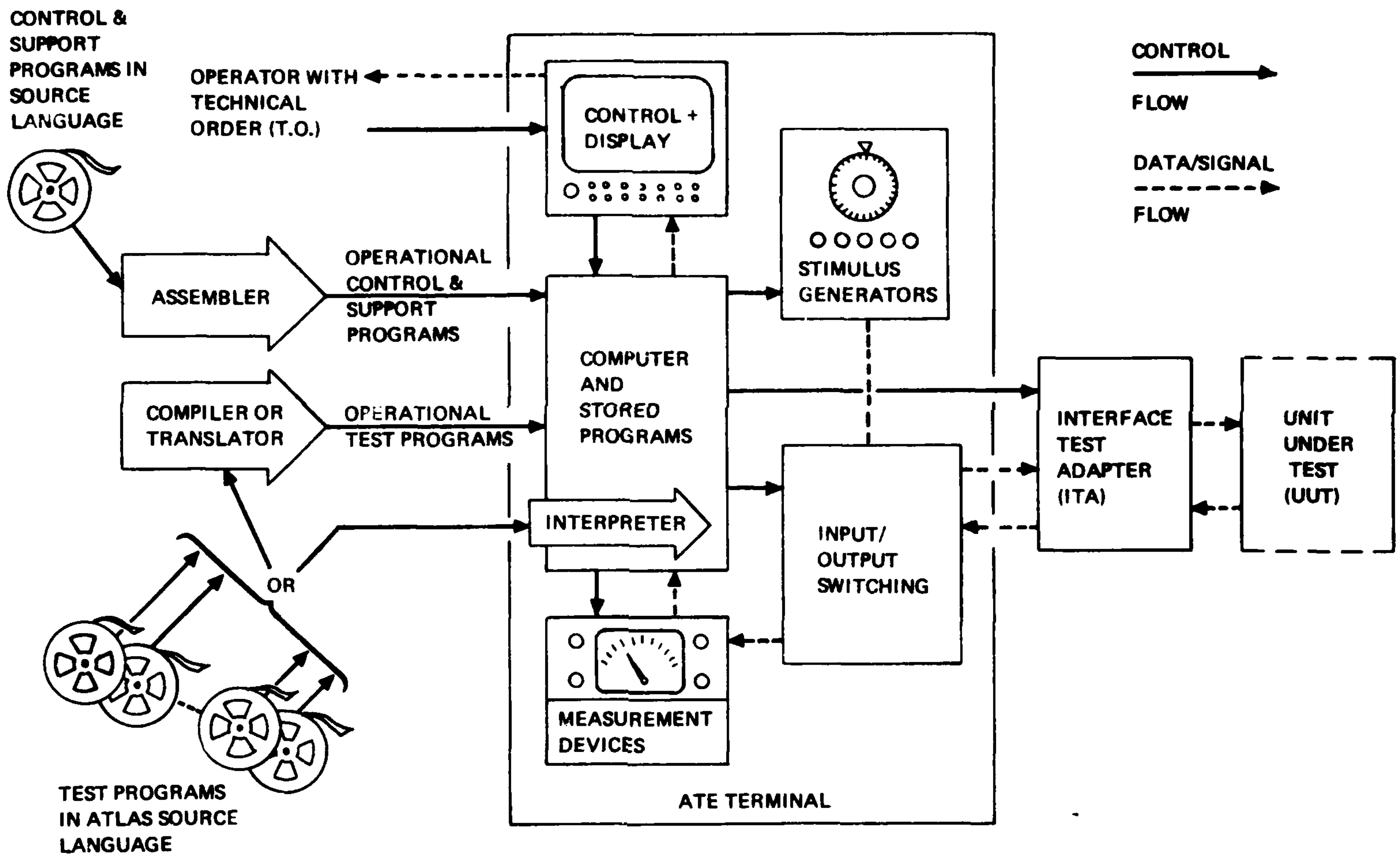


Figure 1.3-2. Typical ATE Configuration

The maintenance level being supported by the ATE is determined by logistics systems analysis.

The importance of the software portion of the ATE system should not be minimized since both the application of the test stimuli and the measurement of the result are achieved via software. Arbitrary function generation and complicated wave analysis can also be accomplished by software.

1.4 GUIDEBOOK ORGANIZATION

The scope and purpose of this guidebook, as well as the general characteristics of TS and ATE systems, are defined in the Introduction (Section 1.0).

Software requirements specification for TS and ATE is discussed in two separate sections: Sections 3.0 and 4.0, respectively. Each of these two sections is subdivided according to major activities/milestones in the process of deriving specifications. This process begins with requirements for the weapon system being supported by TS or ATE and ends with the procurement specifications for hardware and software. The process is specific to each type of ground system and is described in the introduction to each section.

Documents which are most directly applicable to the subject of TS or ATE requirements specification are listed in Section 2.0. Additional supporting documentation is identified in the Bibliography, Section 5.0.

The relationship of guidebook topics to specific paragraphs in government documents is described by a matrix format in Section 6.0. A detailed subject index to the guidebook is provided in Section 9.0.

A glossary of terms and abbreviations/acronyms are provided in Sections 7.0 and 8.0, respectively.

Section 2.0 APPLICABLE DOCUMENTS

The following documents bear directly on the topic of requirements specification for ATE and TS software:

DOD 5000.29, Management of Computer Resources in Major Defense Systems, 26 April 1976

AFR 800-14 Vol. II, Acquisition and Support Procedures for Computer Resources in Systems, 26 September 1975

MIL-STD-483, Configuration Management Practices for Systems, Equipment, Munitions, and Computer Programs, 1 June 1971

MIL-STD-490, Military Standard Specification Practices, 18 May 1972

MIL-STD-1519, Preparation of Test Requirements Documentation, 17 September 1971

MIL-D-83468, Digital Computational System for Real-Time Training Simulators, 12 December 1975

AFLC Regulation 66-37, Management of Automated Test Systems, 24 October 1975

MIL-S-83490, Specifications, Types and Forms, 30 October 1968

MIL-STD-499A, Engineering Management, 1 May 1974

AFR 57-1, Required Operational Capabilities, 30 May 1975

AFM 50-2, Instructional System Development

AFP 50-58, Handbook for Designers of Instructional Systems, Vol. 1-5

AFR 800-2, Outline 1

Section 3.0 TRAINER SIMULATOR SOFTWARE REQUIREMENTS SPECIFICATION

This section describes the software acquisition engineering (SAE) process necessary to derive software requirements for a TS system. This derivation begins with a ROC and concludes with documentation of specific software requirements. The SAE process involves three principal tasks:

a. Technical evaluation (process of deriving software requirements)

b. Planning (definition of TS development approach)

c. Documentation (description/specification of requirements)

Section 3.0 is organized under these principal tasks.

The preparation and issuance of a ROC defines a need for services and/or equipment (hardware/software) to provide a specific TS capability. A ROC defining training simulation needs concerns the training of personnel to operate or maintain a mission vehicle and related equipment. An orderly process is followed for planning and developing an instructional program which insures that crew personnel are taught the knowledge, skills, and attitudes essential for successful job performance. Requirements for a TS - both hardware and software - are specified in the context of facilitating that instructional program.

The Air Force TS software engineer is involved in the process of requirements specification from initial Air Force Systems Command (AFSC) review of a ROC through the last negctiated change to TS requirements (which can occur long after the TS procurement specification is finalized). Emphasis in this guidebook is placed on those activities up to, and including, final approval of the TS requirements specification.

The principal task of the AF TS software engineer in requirements specification is to interpret and augment MIL-D-83468 for the specific TS system being developed. This means tailoring MIL-D-83468, item by item, to match the particular objectives and unique features of the proposed TS. A MIL-D-83468 checklist is provided in paragraph 3.1 to assist this activity.

It is important to note at the outset that TS software requirements cannot be derived independently from TS hardware. TS software and hardware are interdependent and further, the implementation of the TS functional requirements can consider trades between hardware and software capabilities.

Since TS requirements are derived for an integrated hardware/software system, the AF TS software engineer will participate in requirements derivation as a team member. This team, the System Program Office (SPO) cadre and associated consultants, will develop and select a TS design concept which meets user requirements at acceptable cost and risk.

System selection under these criteria often involves the use of off-the-shelf hardware/software components - another reason software requirements cannot be divorced form integrated TS system requirements.

Many major manufacturers of training systems have developed standard modules and high technology software which facilitate their ability to provide simulator systems meeting a wide variety of requirements. This is accomplished by assembling (and providing modifications to) a number of standard hardware and software modules tailored to a specific TS capability. Consequently the manufacturer will have already made hardware/software trades for these modules and

selected an approach which enhances his ability to remain competitive, both in cost and performance. If the TS acquisition engineer places quantitative software performance requirements in the procurement specification, these requirements may negate the contractor's own efforts to achieve cost effectiveness; with the result that the increased cost and associated technical risk necessary to meet these quantitative requirements is passed on to the government. In general, quantitative performance requirements should be specified at the system level, leaving to the contractor such decisions as whether a performance requirement is met by hardware, software, or a combination of these.

Hence, the software requirements determination and specification should not be divorced from system and hardware consideration and the software acquisition engineer is a key part of the SPO cadre. Further, specific software requirements contained in the RFP for TS should generally be limited to qualitative requirements of the type contained in MIL-D-83468. Once the contractor has interpreted the TS requirements in his proposal response to the RFP, then more specific software requirements can be included in the final procurement specification which is negotiated.

The process of developing analysis and data for input to the TS specification is described in paragraph 3.1. This process is explained by the sequence of major events, description of specification activities, and relationship of supporting documentation.

The contents of planning documentation, which supports the requirements derivation process, are described in paragraph 3.2. Then, actual preparation of the TS specification is discussed in paragraph 3.3. This discussion of specification preparation relates how the requirements derivation data/analyses (paragraph 3.1) and planning elements (paragraph 3.2) are used to produce the TS specification.

Two additional paragraphs in Section 3.0 are Problem Areas (paragraph 3.4) and Conclusions (paragraph 3.5).

The flight crew TS is used as the principal example in this guidebook. However, this process of deriving software requirements is generally applicable to other TS (i.e., for other mission stations and maintenance positions).

3.1 TECHNICAL EVALUATION

The term "technical evaluation" is used in this guidebook to describe the process for developing software requirements for TS systems. Figure 3.1-1 illustrates, in general, how TS software requirements evolve from a ROC. The progression is from the ROC to the TS system, from the TS system to the computational system and the allocation of requirements to hardware and software within the computation system. In actual practice, the flow is not always direct. There are iterative paths and interdependencies between "levels" of requirement specification.

The goal of technical evaluation is to develop supporting data for TS requirements that are technically feasible, responsive to ROC requirements and within cost constraints. These supporting data are then utilized for preparation of TS procurement specifications (paragraph 3.3).

The process of TS software requirements derivation is described in two principal ways:

a. The sequence of major events (milestones) leading up to final approval of the TS requirements specification, and

b. The major derivation activities associated with those events.

This description of the process is organized in this section under the major derivation activities - as shown in Figure

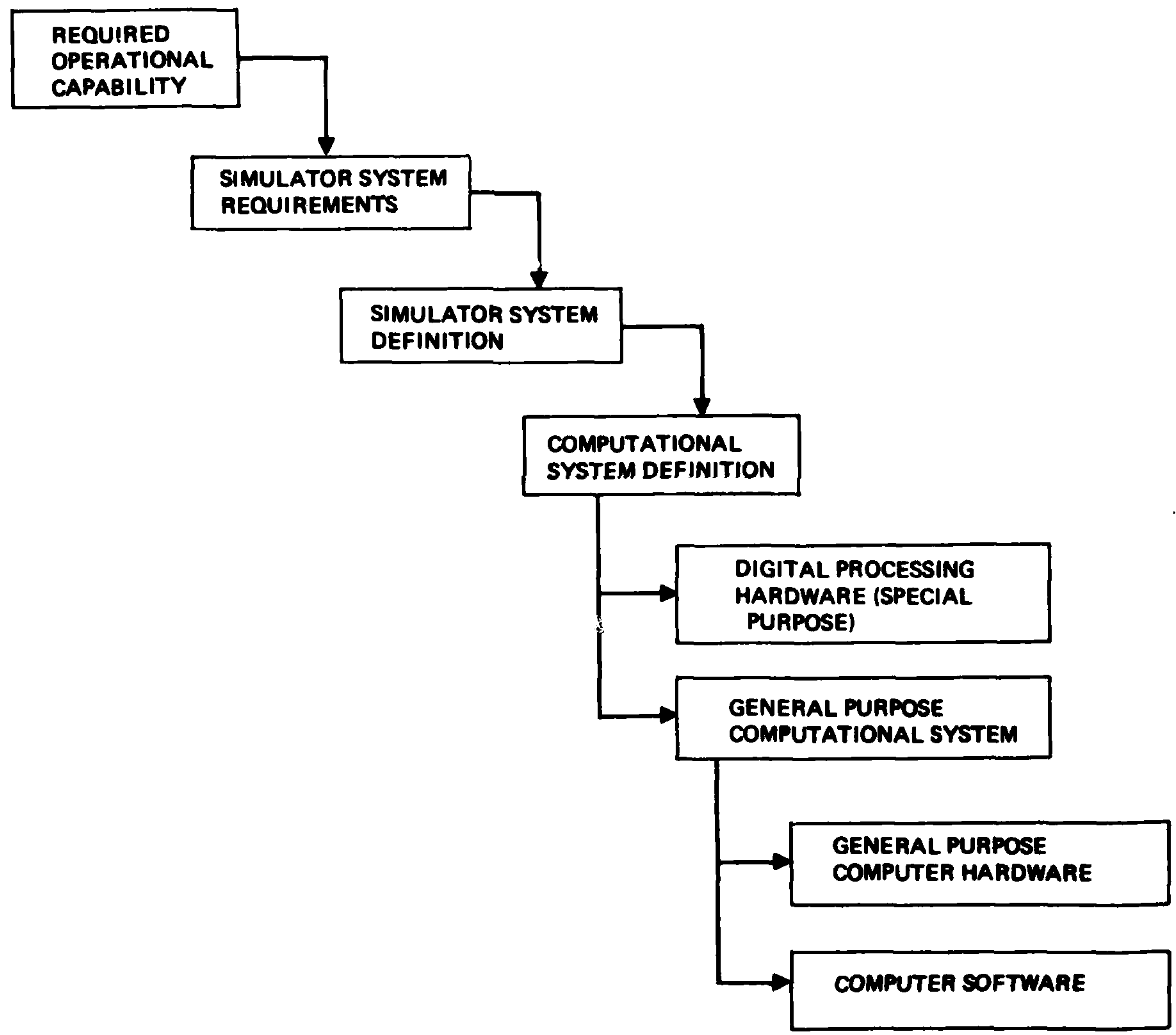


Figure 3.1-1. Evolution of TS Software Requirements

3.1-2. Also shown in the figure are organizations having principal responsibilities for the identified activities. Each activity is described in the indicated section paragraphs.

The sequence of major events in TS requirements derivation is illustrated in Figure 3.1-3. These are then tied to derivation activities in Figure 3.1-4. The figure provides a good overview of the requirements derivation process and warrants careful inspection. Arrows between the boxes in the diagram signify information exchange, for example, the definition of candidate TS systems depends on inputs from (1) definition of training simulation requirements, (2) TS systems analyses and trades, and (3) TS preliminary design. Also, some relationships are double-arrowed, for example between "TS systems analysis and trades" and "TS preliminary design". This means the process is iterative and neither activity is completed until both are completed. Each of the primary activities is discussed in paragraphs 3.1.1 through 3.1.6.

The relationship of supporting documents to the process events and activities is shown in Figure 3.1-5. Not all relationships between process elements are shown in Figure 3.1-5 (additional lines and arrows would clutter the figure) but principal relationships are indicated. Figure 3.1-5 provides a composite view of elements in the requirements derivation process and, although the diagram is somewhat involved, the process is rather straightforward when each activity/event is considered individually (in the following paragraphs).

As a further aid to tracking the process of TS requirements specification, a detailed checklist of specific events was developed. This checklist is given in Table 3.1-1. Significant events in the requirements specification process are arranged in their chronological order. The table also has a column to record the planned completion date, the current status and the date the event is completed. The table can be used as a convenient checklist for planning and evaluating the requirements derivation. Not all TS requirements developments will follow this exact sequence of tasks but the checklist can be modified to suit different development approaches. This checklist is a composite of two AF procedures. Items 1 to 5 are from AFR 800-2, outline 1. Items 6 to 11 result from a review of AFP 50-58 (Handbook for Designers of Instructional Systems).

3.1.1 ROC Review

The ROC is examined to discern required TS system characteristics, mission objectives and functions, and minimum acceptable system-level technical performance requirements. The following are examples of TS functions:

a. Simulate selected on-board systems operations

b. Simulate weapon system physiological environment

c. Simulate weapon system operation environment

d. Provide instructor control features

e. Provide advanced instructional features

Simulation is an approximation or representation of real world phenomena. A successful training simulation is one in which the student perceives "realistic" sensory inputs and system responses; at least with sufficient fidelity to prepare the student for actual operational situations. Additional criteria for a successful TS are that the TS provides (1) the range and diversity of situations associated with crew personnel duties and mission tasks, (2) feedback to the student as rewards/penalties for specific behaviors, and (3) instructor monitoring and evaluation of trainee performance.

Technical performance statements in the ROC may be stated either qualitatively

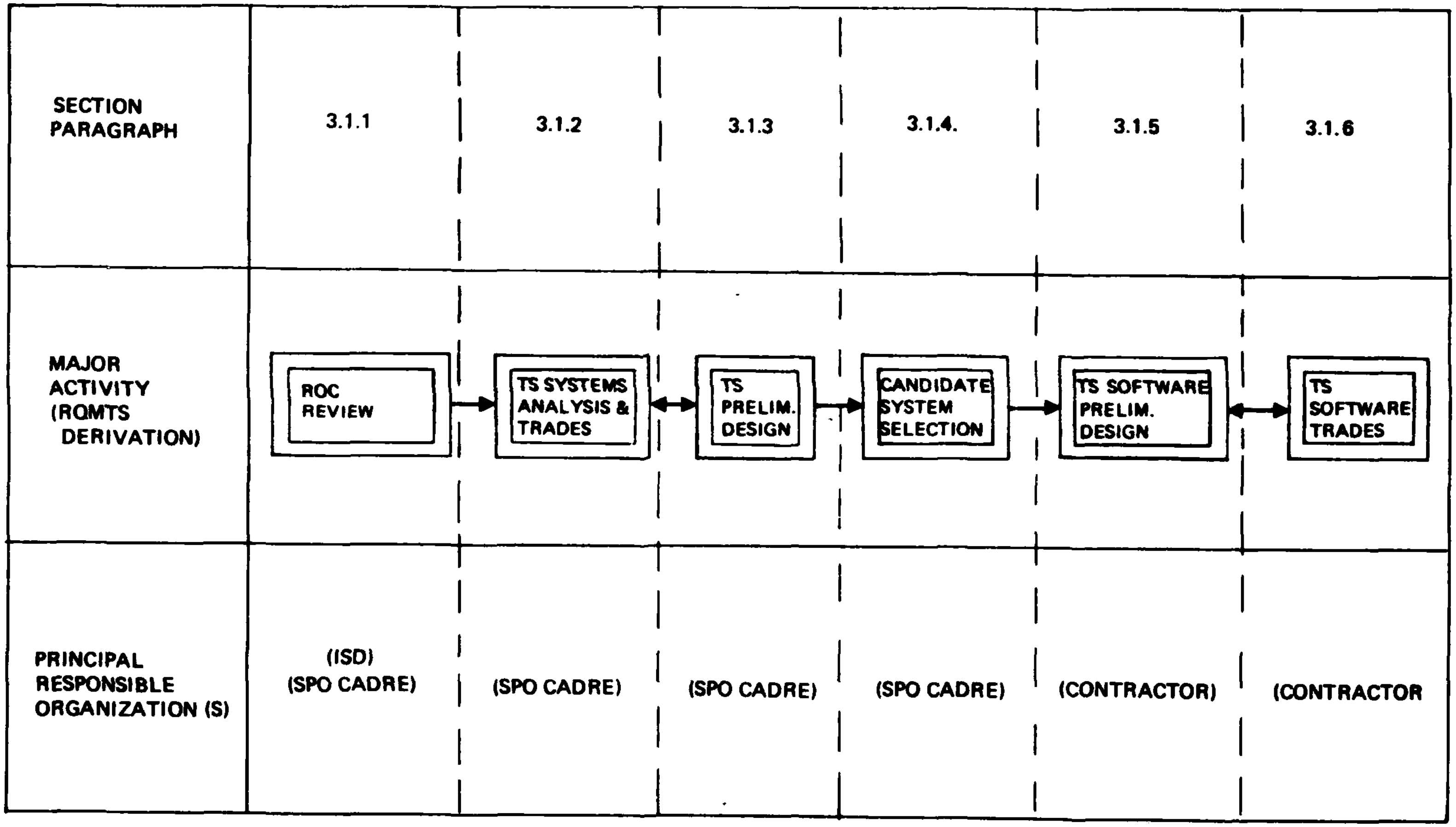


Figure 3.1-2. Requirements Derivation Activities

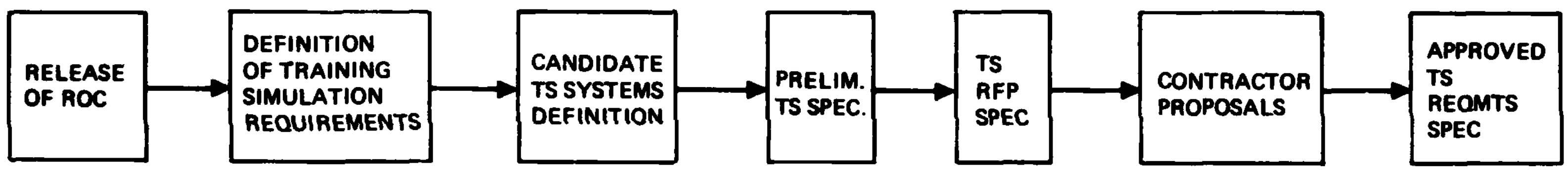
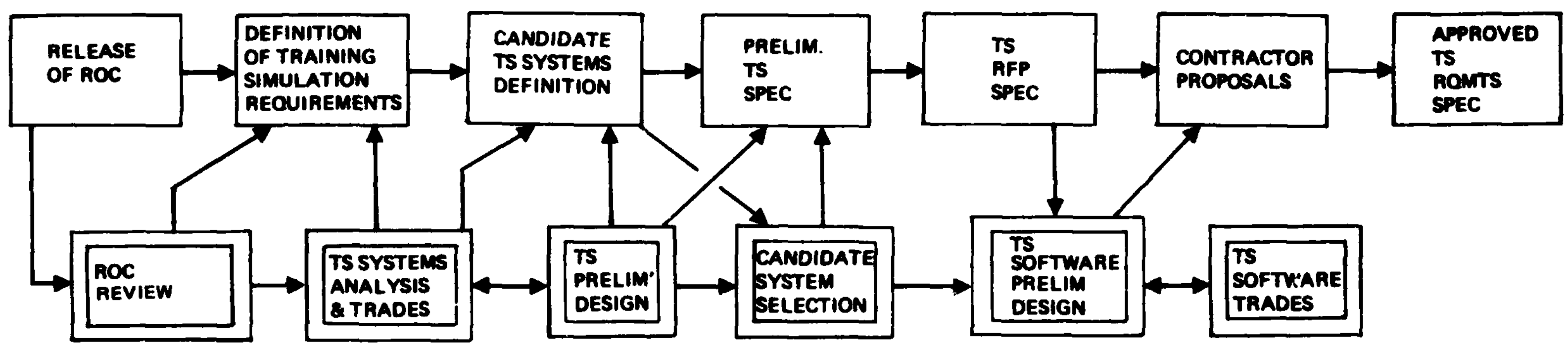


Figure 3.1-3. Major Events in TS Requirements Derivation

EVENT MILESTONES IN DERIVATION PROCESS



REQUIREMENT DERIVATION ACTIVITIES

Figure 3.1-4. Relationship of Derivation Activities to Major Events

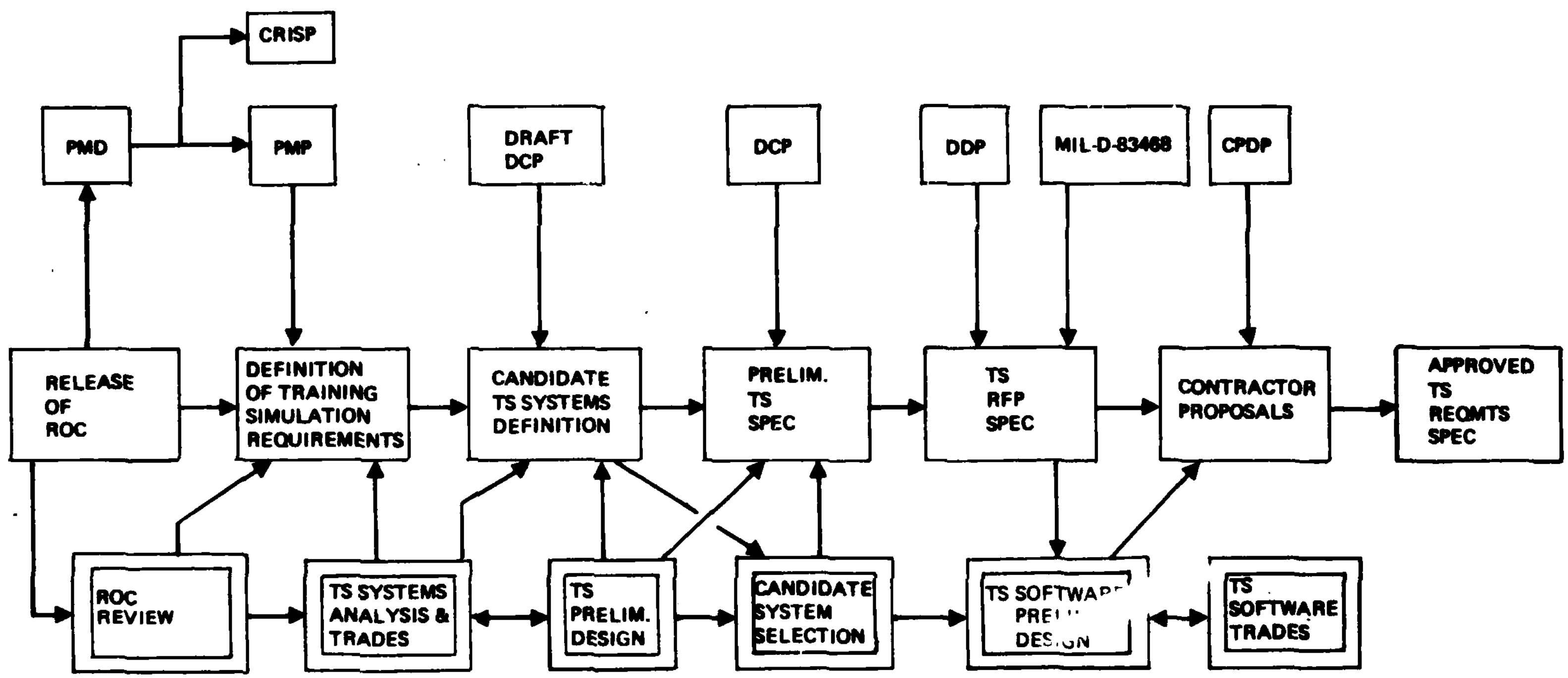


Figure 3.1-5. Composite of Requirement Process Elements

Table 3.1-1. Requirements Specification Event Checklist (Sheet 1 of 3)

EVENT	PLANNED COMPLETION	STATUS	DATE COMPLETED
1. USING COMMAND SUBMITS ROC TO HQ USAF			
2. HQ USAF DISTRIBUTES ROC TO USAF AGENCIES			
3. AGENCIES REVIEW AND RETURN ROC TO HQ USAF			
4. HQ USAF APPROVES ROC AND ISSUES PMD			
5. ASFC DIVISION FORMS A SPO CADRE			
6. STUDY EFFORT TO DETERMINE MEANS TO SATISFY ROC			
a. STUDY GROUP RECEIVES ROC AND PMD			
b. T.I. MEETING TO DEFINE STUDY OBJECTIVES AND PRODUCTS OF STUDY			
SPECIFIC EFFORTS INCLUDE:			
• ESTABLISH TRAINING/SIMULATION OBJECTIVE			
• ESTABLISH STUDY SCHEDULE			
• DETERMINE JOB SYSTEM PERFORMANCE REQUIREMENTS			
• DETERMINE THE TRAINING/SIMULATION EQUIPMENT (HW & SW) REQUIREMENTS			
• REVIEW SYSTEM EQUIPMENT			
• PREPARE LIST OF TRAINING EQUIPMENT (HW & SW) AND PRELIMINARY DESIGN LAYOUT			
• PREPARE DEVELOPMENT CONCEPT PAPER (DCP)			
c. COMPLETE TASK/FUNCTION DESCRIPTION WORKSHEETS			
d. COMPLETE CRITERION OBJECTIVE AND TRAINING/SIMULATION REQUIREMENTS WORK SHEETS			
e. ESTABLISH TRAINING REQUIREMENTS			

Table 3.1-1. Requirements Specification Event Checklist (Sheet 2 of 3)

EVENT	PLANNED COMPLETION	STATUS	DATE COMPLETED
6. f. COMPLETE TRAINING/SIMULATION MEDIA TRADE STUDY			
g. COMPLETE SURVEY STUDY OF TRAINING/ SIMULATION SYSTEMS AND METHODS			
h. T.I. MEETING			
● PRESENT PRELIMINARY OUTLINE OF DCP			
● FINALIZE CRITERION OBJECTIVES AND RECOMMENDED TYPE OF TRAINING/ SIMULATION MEDIA			
● ESTABLISH STUDY'S TRAINING/SIMULATION BASELINE TO BEGIN EQUIPMENT SELECTION			
● ESTABLISH GUIDELINE FOR SPECIFICATIONS			
i. COMPLETE PRELIMINARY EQUIPMENT (HW & SW) SELECTIONS			
j. COMPLETE PRELIMINARY DRAFT OF DCP			
7. SUBMIT DCP FOR REVIEW			
8. DETERMINE METHOD OF SPECIFICATION			
● SYSTEM SPECIFICATION			
● HARDWARE SPECIFICATION			
● SOFTWARE SPECIFICATION			
9. DEVELOP DRAFT REQUIREMENTS SPECIFICATIONS			
● SYSTEM SPECIFICATION			
● HARDWARE SPECIFICATION (IF REQUIRED)			
● SOFTWARE SPECIFICATION (IF REQUIRED)			

Table 3.1-1. Requirements Specification Event Checklist (Sheet 3 of 3)

EVENT	PLANNED COMPLETION	STATUS	DATE COMPLETED
10. T.I. MEETING			
REVIEW DCP			
REVIEW DRAFT SPECIFICATIONS			
DRAFT SOW			
11. SUBMIT VISIBILITY SPECIFICATIONS AND DCP			
12. ISSUE RFP			
13. COMPLETE SOURCE SELECTION (SELECT CONTRACTOR)			
14. SUBMIT FINAL REQUIREMENTS SPECIFICATION			
15. PLACE CONTRACTOR PROPOSAL UNDER CONTRACT			

or quantitatively. For example, the ROC may state that the trainer is to be like those used with a given aircraft by commercial airlines. This is a qualitative indication to TS engineers of the scope and nature of the system the users have in mind for the trainer, even though it is not a wholly definitive one.

At this point in the evaluation, there is no attempt to differentiate between hardware and software functions or subfunctions, except for those which may have been explicitly stated in the ROC. However, the characteristics, objectives, etc., which are included in the ROC, need to be examined for feasibility and attainability with respect to current trainer technology, physical resources, human (trainer and instructor) performance capabilities, life cycle costs, and other constraints. The simulator engineer(s) has signficiant contribution to make relative to this task, based on his experience with other TS systems.

As noted in Figure 3.1-4, the ROC review activity provides input to (1) definition of training simulation requirements and (2) TS systems analyses and trades. This activity is conducted by two principal organizations (Figure 3.1-2): an AF Instructional System Development (ISD) team and the SPO cadre. The major task of this joint effort is to begin defining training simulation requirements.

Both the SPO personnel subsystem/training equipment manager, and designated USAF agency training equipment manager will be potential co-chairman of the SPO cadre. The cadre will usually consist of the designated USAF agency coordinator and AF subject matter specialists, designated specialists in training services and equipment, representatives of the weapon system using command and other AF agencies as required. Weapon system and TS contractor personnel are also key participants.

Procedures for analysis of training objectives and requirements are those inherent in the ISD process. The ISD process is described in AFM 50-2 and the Handbook for Designers of Instructional Systems, AFP 50-58.

3.1.2 TS Systems Analysis and Trades

The technical evaluation process includes various analyses and trade-offs to translate the overall system requirements statements in the ROC to a specific set of requirements for the TS system (Figure 3.1-4). Analysis and trade-off techniques are employed to select a set of requirements for a system that can be produced within allowable costs, has low technical risk, is within current state-of-the-art and is responsive to user needs. This activity depends on inputs from the ROC review and from TS preliminary design. As noted in Figure 3.1-4, this activity is interactive with TS preliminary design.

Discussion of TS systems analysis and trades is approached from two directions:

a. How ROC-derived requirements and preliminary design interact to establish training system requirements, including software requirements.

b. Examples of representative analyses and trades.

These are treated separately in the next two sections.

3.1.2.1 Training System Requirements for Flight Simulators. The experience of flight in training simulation can range from a minimum of Horizontal Situation Indicators (HSI) and Attitude Direction Indicators (ADI) for the pilot to a maximum of out-of-the-window view, cockpit motion, audio cues and fully operational cockpit controls.

The basic hardware components of a flight simulator are a computer, a cockpit and an interface. Selection of refinements such as a motion base, a visual display system, audio cues and the

instrument/control complement will scope the detailed interface configuration and specific software requirements.

The total system includes computer peripherals for input/output capabilities and utility software to be used in software development and operations.

Model requirements for a TS system can be derived from the required sensory inputs to the student, as specified or interpreted from the ROC. For the case of a flight simulator, the major categories of input are:

a. Cockpit displays

b. Visual display

c. Motion

d. Audio cues

e. Control loading

These items translate into hardware and software components and Table 3.1-2 provides an example of such a translation.

Real-time software packages to support flight simulation are flow charted in Figure 3.1-6. Usefulness of the simulator as a training tool is facilitated by means of instructor-interactive software for malfunction insertion/deletion; flight condition selection; mode control and other features. The executive program (Figure 3.1-6) working input/output (I/O) routines and interrupt handlers provide trainer controllability.

The balance of the software system is made up of non-real-time processors, utilities, and diagnostics which provide training flexibility and maintenance capabilities. Processors are assemblers and compilers. Examples of utilities are source edit programs, link-loaders, file-merge/delete routines, dump routines and debug packages. Diagnostics are programs which exercise hardware, usually by causing information transmittal through critical interfaces. Examples of diagnostics are:

a. Micro level checking of computer Central Processing Unit (CPU) capability to executive instructions

b. Checking memory

c. Exercising I/O interfaces, checking status indicators, parity

d. Exercising standard peripherals, checking peripheral response to control and data

e. Exercising external interfaces and simulator-unique hardware, checking signal returns, indicators, and physical displacements

Simulation hardware requirements are both general purpose (a digital computer, its peripherals, analog/digital, digital/agalog converters and discrete lines) and special purpose (interface logic and drivers for the cockpit display and control inputs). In the same way, software requirements exist for off-the-shelf processors - assembler, compiler, utilities - and for programs specially written to model the particular airplane subsystems.

Software requirement derivation and hardware requirement derivation are processes that interact with each other.

3.1.2.2 Examples of Analyses and Trades. Following ROC evaluation, a complete list of the specific TS system functions and subfunctions is derived. Some may be in the ROC, but others may need to be established by additional coordination or analysis. Representative trainer mission functions are described below alone with examples of subfunctions that could be associated with each one.

a. <u>Simulate Selected On-Board Systems Operations</u> - The subfunctions which may be included depend partly on the type of weapon system which is involved, but examples are:

Table 3.1-2. Real Time Model Software Items

	Hardware Required		
Function	Interface	Cockpit	Software Required
Cockpit displays	Digital/Analog converters, Discrete outputs Synchro outputs Discrete inputs	Pilots' and Flight Engineers instruments, gauges and lights	Models for each flight subsystem; engines hydraulics, electrics, radio aids
Visual display	Digital/Analog converters, synchro outputs, discrete outputs	Image acquisition, projection equipment	Algorithms to produce drive to image acquisition equipment, given translational, rotational parameters from airplane dynamics solution
Motion	Digital/Analog conv., Analog/digit. convert, discrete outputs	Hydraulically driven motion base hydraulic power supply	Algorithms to produce drives to hydraulic-control amplifiers given transl., rotational description parameters from airplane dynamics solution
Audio cues	Discrete outputs, digital/analog converters	Sound synthesizers, amplifiers, speakers	Algorithms to drive synthesizers given status of subsystems related to sound sources; engines, hydraulics, etc.
Control loading	Digital/analog, analog/digital converters	Control-drive hydraulics, amplifiers	Data for flight condition, load factor, hinge moments and blowdown limits.

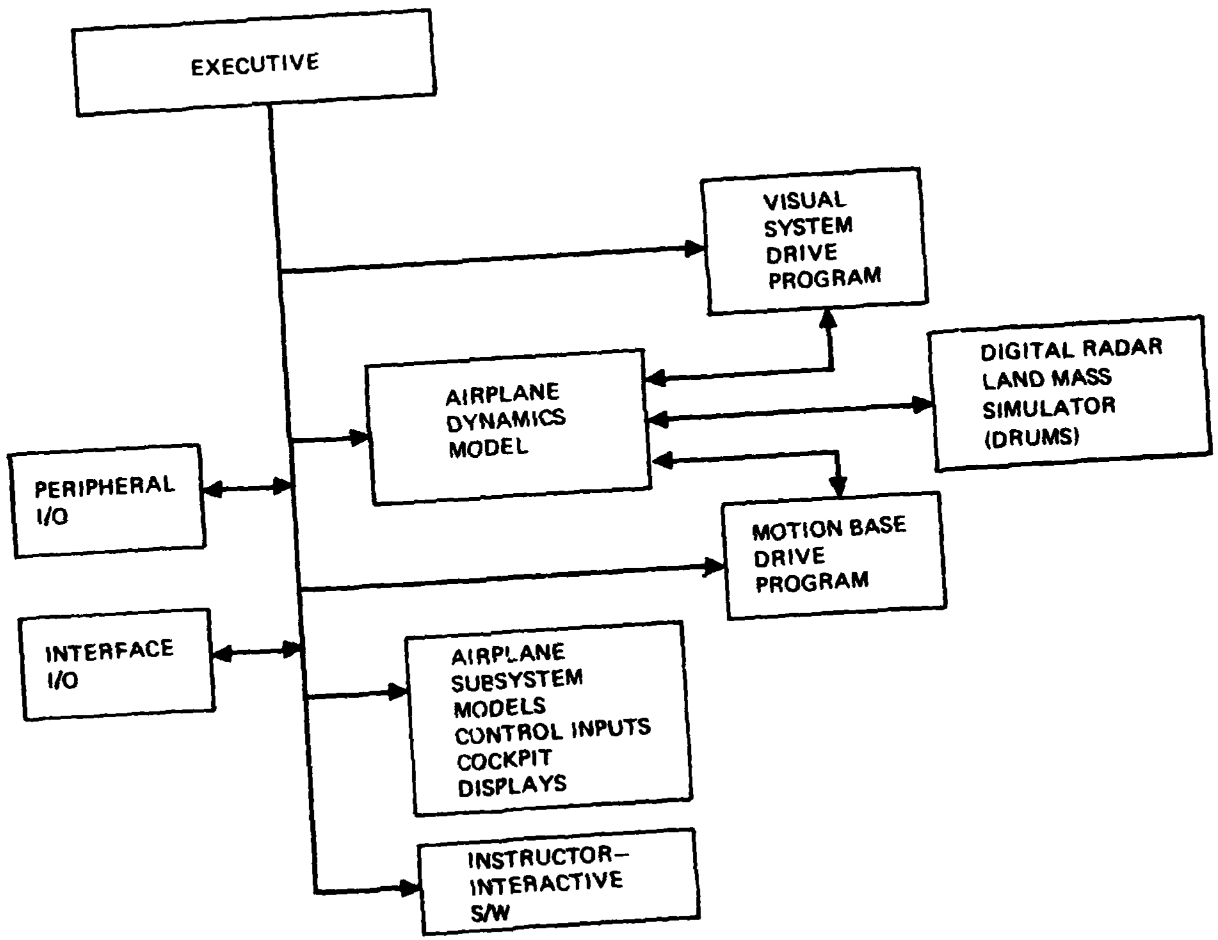


Figure 3.1-6. Block Diagram—Real-Time Flight Simulator Software

(1) On-board weapons systems such as air-to-air and air-to-ground missiles, gravity weapons, cruise missiles and rockets

(2) Flight control system

(3) Communications

(4) Flight instruments

(5) Navigation

b. Provide Instructor Control Features - Instructor control is affected by factors such as simultaneous instructor control of multiple trainee positions and the number of instructor positions. Another factor is the instructor's requirement to be able to override or reset processes initiated by trainees. In addition, there is a host of stimuli and conditions which the instructor may have to control at each position.

c. Provide Advanced Instructional Features - Several automatic features may be specified:

(1) Provide automatic control of initial conditions

(2) Provide automatic demonstration

(3) Provide automatic malfunction insertion

(4) Provide automatic monitoring of procedural items

(5) Provide automatic permanent recording of results

(6) Provide student's feedback capability

(7) Provide automated performance comparisons.

Examples of three specific analyses are:

a. Visual Display Tradeoff - A visual display system is required. If the simulator is planned for night flight training, could one of the computer-generated CRT displays suffice? Note that in going to a CRT-type display, a peripheral mini computer is probably needed for refresh of the CRT image. Appropriate software to interface between the simulator computer and the mini is needed. In trade for this added cost is relief from the electromechanical complexity of image generation using a moving television camera.

b. Life Cycle Cost (LCC) Analysis - The subject of LCC is discussed in the Cost Measuring and Reporting SAE Guidebook. A brief summary of this discipline is provided herein, both as an example of an important analysis and because LCC appears as a parameter in other studies. Experience has shown that early decisions in system concept and definition phases have the greatest potential for cost savings. Experience utilizing current LCC financial reporting techniques, in particular the Air Force Logistics Command (AFLC) model for logistic support costs, enables the implementation and utilization of an effective LCC program to assess TS software during this critical point in development. Cost drivers are defined, challenged, and trade studies made to reduce the impact of the cost drivers on the support costs. Cost-avoidance disciplines, including design simplification and optimum use of standard modules should be stressed early in the formulation of TS requirements. The 20-year LCC analysis also provides the means for establishing cost targets, monitoring acquisition costs, and instituting corrective action.

The LCC process is summarized in Figure 3.1-7. A baseline system for the TS is initially established from which trade studies are conducted on various alternatives. The baseline requirements are determined, assessment of the reliability (R) and maintainability (M) made, and logistics support analysis (LSA) of the TS is performed, based on MIL-STD-1388.

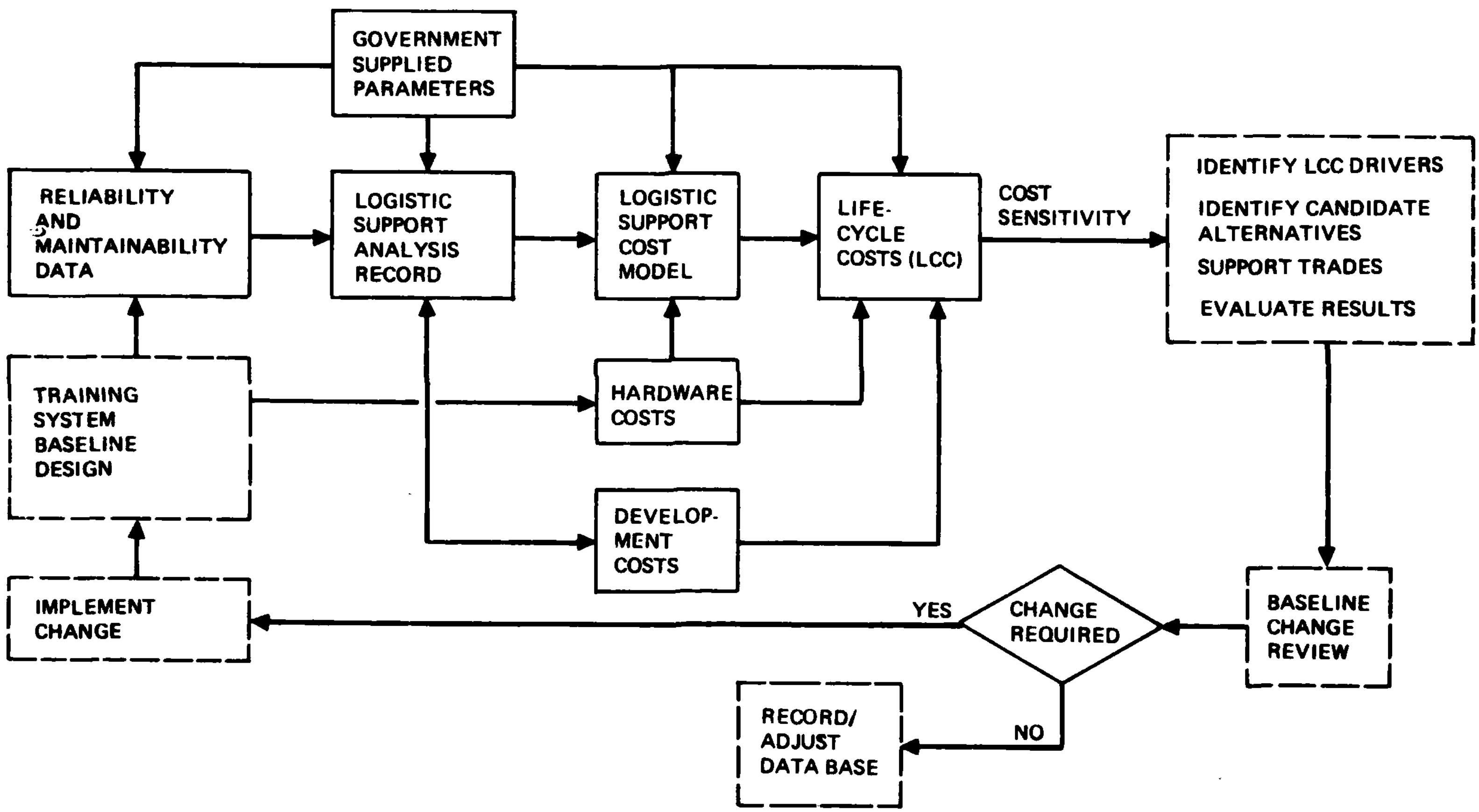


Figure 3.1-7. Life-Cycle Cost Analysis and Design Interface

The Logistics Support Costs (LSC) model input data includes parameters such as repair cycle times and labor rates. This data, together with the Logistics Support Analysis (LSA) data and the unit costs, complete the LSC model data requirements.

The LCC of the baseline TS concept is then determined from the LSC, acquisition costs and development costs. The LSC data is used in trade studies to identify cost drivers and candidate alternative approaches are measured in the model. Full consideration is given to adopting existing hardware and software for use in reducing LCC. Other systems in being, or planned for activation, are reviewed and approaches evaluated for use of common support requirements to the extent permitted by the development concepts. The overall trade study process is a multiple-disciplined effort involving procurement engineering, test and logistics disciplines. After each trade study has been complete, a detailed evaluation of the results is performed and changes to the TS baseline evaluated.

c. Risk Management - A key element is the requirement specification for the TS system and its risk assessment.

As previously stated, a principal goal of the configuration engineer should be to minimize risk to the maximum extent practical, consistent with supporting requirements reflected in the ROC and its supporting documentation. Such factors as the existence of "off-the-shelf" or easily-modifiable software and hardware is a significant factor affecting both schedule and cost. Figure 3.1-8 and Table 3.1-3 illustrates a formal means whereby high risk items are identified and continually reviewed while risk abatement action is taken. Also presented are example criteria for judging whether any particular risk item is of sufficient magnitude to treat in this manner. Risk assessment is a continuing process and normally is reported at periodic program reviews. For risk assessment to be effective, a clear definition of what constitutes a risk must exist. Table 3.1-3 identifies criteria for making this judgment. When a problem area has been identified, it must be judged to rate "low" for technical, schedule, and cost for it to be rejected as a risk item.

3.1.3 Preliminary Design of TS

While preliminary design of a TS will most likely be accomplished by engineering specialists in the contractor's organization, the Air Force software engineer will be involved in at least a monitoring and evaluation role. This section describes how TS requirements are derived during the preliminary design activity, with particular emphasis on software requirements.

Preliminary Design (PD) is a high-level treatment of the simulator configuration allowing major interfaces to be identified along with functional elements responsible for the main operating capabilities required. Functional elements are both hardware and software and must be considered together. General design requirements on TS software evolve in the PD process. An example of a highest-level design for a flight simulator is shown in Figure 3.1-9. This diagram might result from a ROC specification which identified:

a. Visual display

b. Motion base

c. Operational cockpit

d. Instructor console

The block diagram in Figure 3.1-9 shows the interface relationship between the various functions and the computational system. Once the TS function and their interfaces are defined, the computational system can be defined. The general size and capability of the simulation computer(s) can be established by comparison with previously developed systems and applying the appropriate

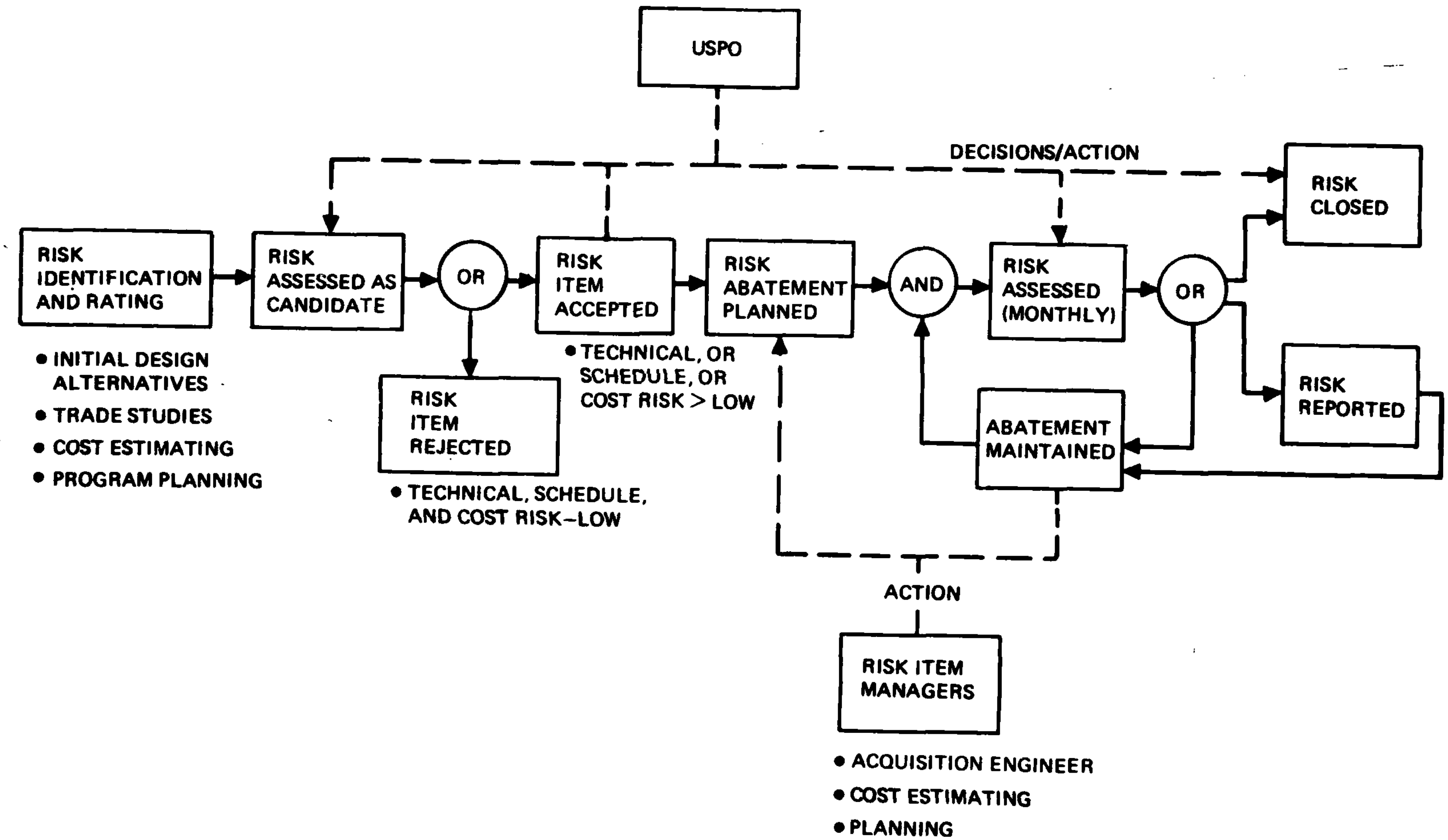


Figure 3.1-8. Risk Analysis

Table 3.1-3. Program Risk-Rating Criteria

Risk Rating		Schedule Variance (S_V)	Cost Variance (C_V)
High	• Requirements not defined • Major state-of-art advancement • Capability inadequate • Serious problems - no solution • Expect to fail to meet requirements • Program management action mandatory	$>$ 10% of time remaining	$>$ 20%
Moderate	• Requirements ill-defined/late/changing • Some state-of-art extension • Capability marginal • Significant problems - have potential solution • May fail to meet requirements • Alert program management	Between 8% and 10% of time remaining	$15\% < C_V < 20\%$
Low	• Requirements well-defined • Technology in-hand • Capability adequate • No significant problems • Will meet requirements through normal development • No action required	$<$ 8% of time remaining	$<$ 15%

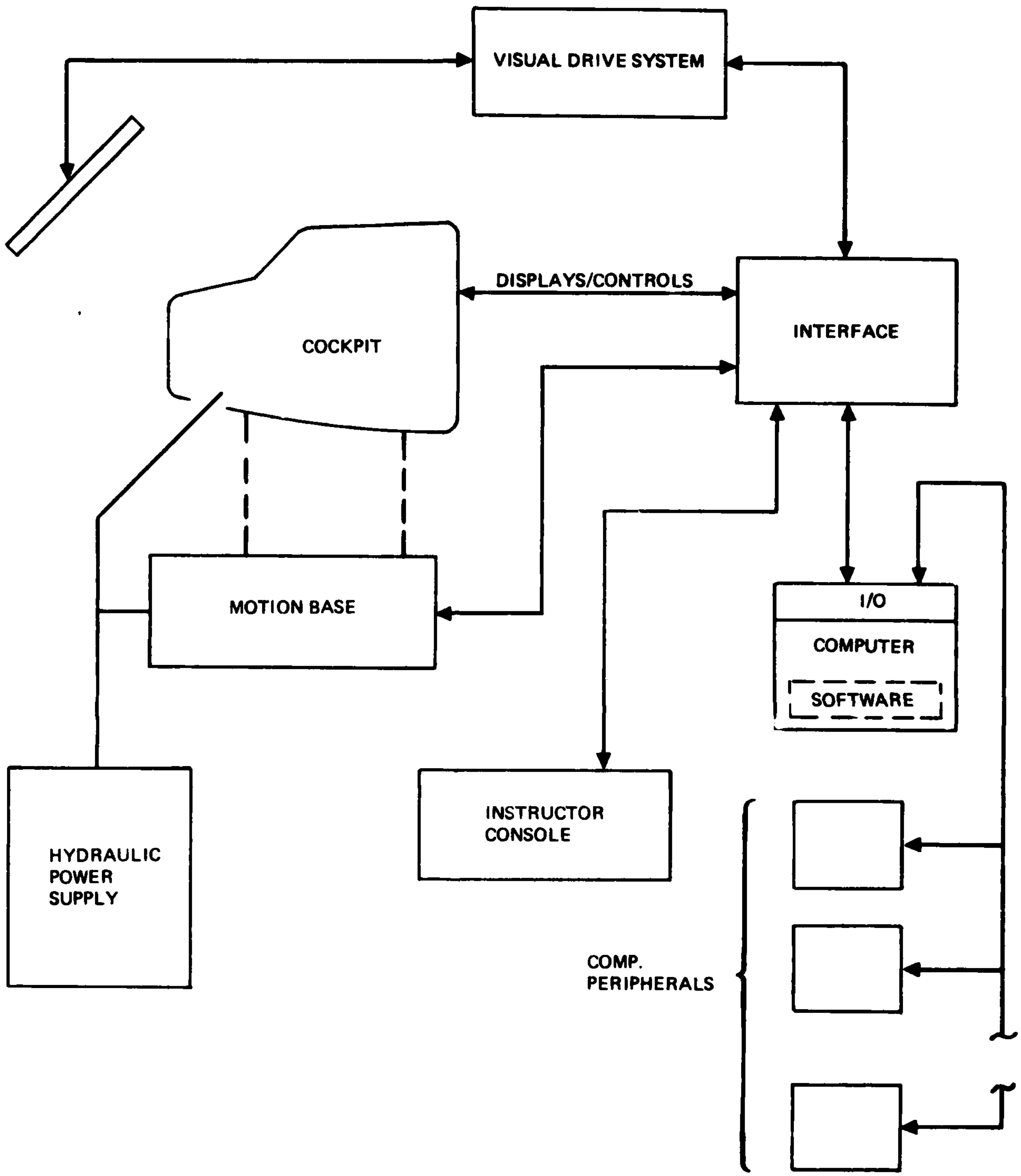


Figure 3.1-9. Block Diagram—Flight Simulator

factors. The general types of computer programs can also be defined such as:

a. Real-time operational programs

b. Simulator support programs

c. Computer program system support programs

d. Maintenance and test programs

e. Calibration test programs

Preliminary software design results from expanding on the basic software items needed to drive such a simulator. The basic items are:

a. Airplane dynamics model

b. Airplane subsystems models

c. Motion base drive program

d. Visual system drive program

e. Instructor - interactive software

f. An executive to manage program interaction sequence and input/output through

(1) Peripheral I/O

(2) Interface I/O routines

These are the basic real-time software items only. The ROC for the instructor station might include, for example, display CRT, mode control and malfunction insertion/deletion capability. Item (e), the Instructor - interactive software could be depicted as in Figure 3.1-10.

Suppose further that the CRT is required to display status of the simulator: malfunctions inserted, flight condition, flight subsystem status and other status data. A data base of model parameters, control inputs, flag, etc., would be the logical source for data acquisition. The general interface diagram shown in Figure 3.1-10 implies I/O software and intercept handling. Adding these additional capabilities produces a software preliminary design as shown in Figure 3.1-11.

Figure 3.1-11 is by no means detailed, but at this point a software systems engineer can begin to make reasonable approximations of the size and complexity of the software segments involved. Character-decoding, I/O routines and interrupt handlers are well known; "acquire variables for display" is an unknown and probably a large programming task. System trades and their software impact can now be made using such a preliminary design.

To assure that all software and hardware requirements are included in the preliminary design, check lists were devised. These are provided in Table 3.1-4 and Table 3.1-5 for software and hardware, respectively. Applicable paragraphs in MIL-D-83468 are also noted for each TS software function in Table 3.1-4.

Based on the training/simulation requirements and TS preliminary design, the SPO cadre will define several candidate TS systems (Figure 3.1-4). The type and number of candidate systems is influenced basically by the background and experience of cadre members and augmented by media/equipment surveys.

3.1.4 Candidate System Selection

Once several candidate TS systems are defined (as noted in the previous paragraphs), the SPO cadre will proceed with selection of one candidate system on which to formulate a preliminary TS requirements specification (Figure 3.1-4).

One method for evaluating alternate system configuration is discussed below. This method identifies criteria categories for evaluating alternate TS configurations, applies a weighing factor for each category and compiles the results in a decision table in which the results can be quantitively evaluated. This technique can be employed at each

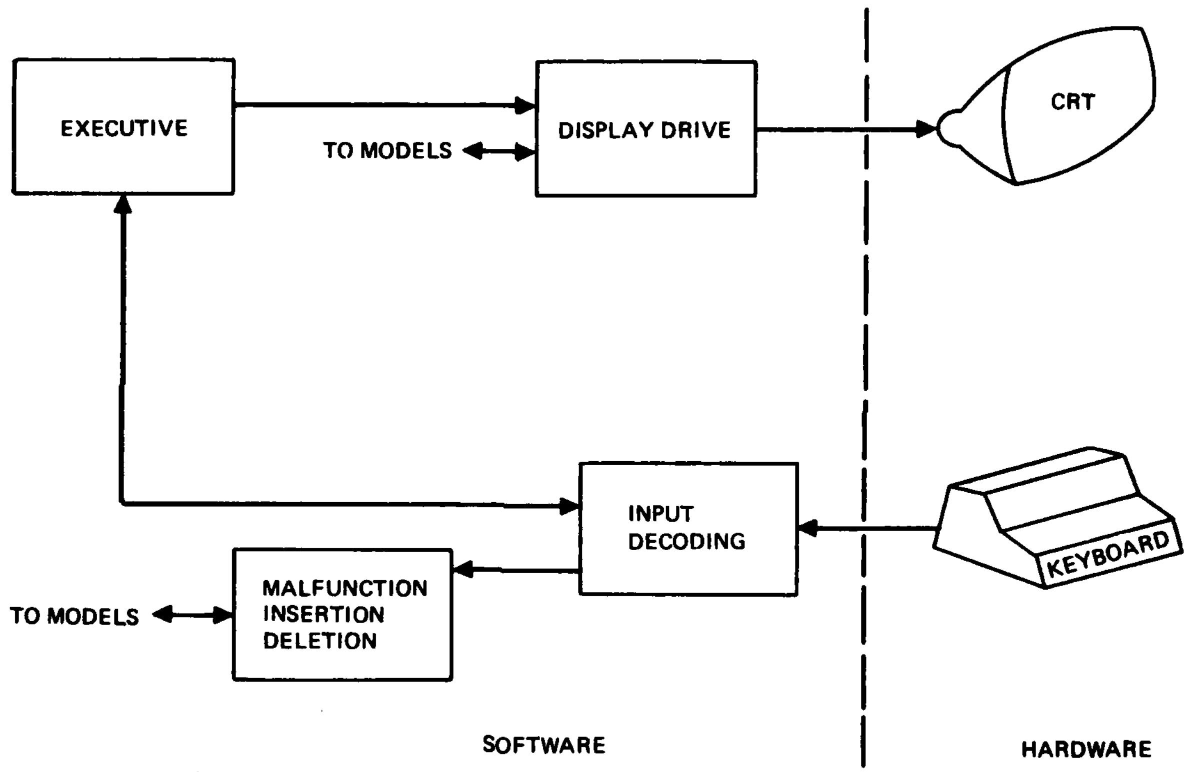


Figure 3.1-10. Instructor—Interactive Software

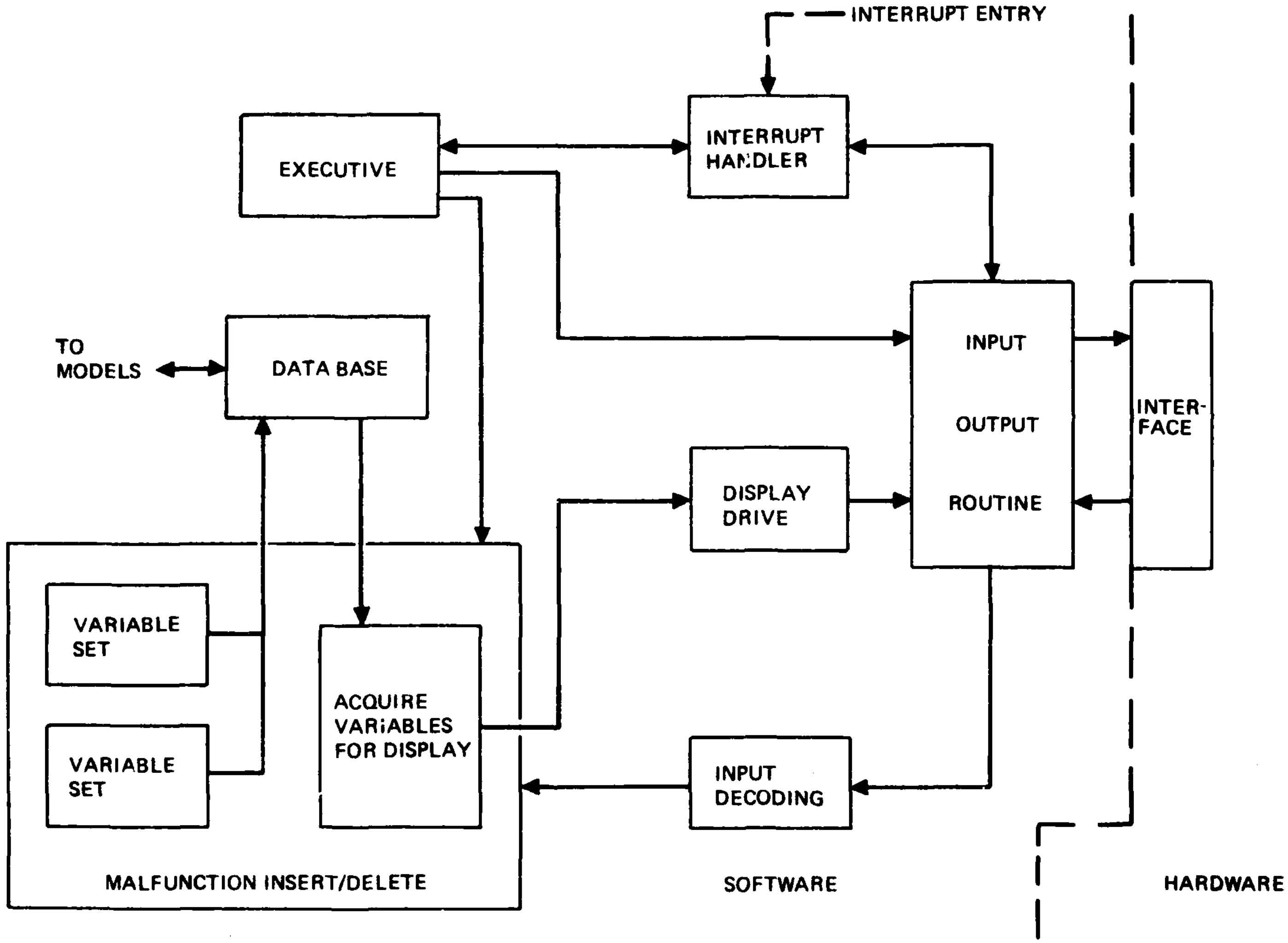


Figure 3.1-11. Preliminary Design Instructor—Interactive Software

Table 3.1-4. Checklist for Software Requirements (Sheet 1 of 3)

Category	Function	Software Item	Applicable Paragraphs MIL-D-83468
Models	Airplane dynamics	Aerodynamic Data Equations of Motion Transformation Material Integration Subroutine	3.2.1, 3.2.2, 3.2.2.2 3.2.2.2.1, 3.2.2.2
	Motion base drive program	Algorithms to generate hydraulic jack signals	3.2.2, 3.2.2.2
	Visual system drive program	Matrices to convert translation and rotational parameters to drive signals for image acquisition	3.2.2, 3.2.2.2
	Audio effects	Sound synthesizer control	3.2.2, 3.2.2.2
	Airplane subsystems	Engines Hydraulics Electrics Cabin Conditioning Radio Aids Navigation Control Loading Fuel Management	3.2.1, 3.2.2, 3.2.2.2
	Interface drive - Cockpit instruments - Control inputs from cockpit	Digital/Analog, Analog/Digital buffers, scaling, I/O discretes, Digital data buffers	3.2.2, 3.2.2.2
Instructor-interaction	Mode control	Interrupt handler, Exec.	none
	Flight condition	Input character decode from keyboard, interrupt handler	none

Table 3.1-4. *Checklist for Software Requirements (Sheet 2 of 3)*

Category	Function	Software Item	Applicable Paragraphs MIL-D-83468
	Malfunction insert/delete	Input character decode, interrupt handler; data base, data base manipulator	None
	Core access	Core access debug, breakpoint, set/clear	3.1.4.1.2
	Display status	Data base acquisition routine	None
	Drive & refresh CRT by means of auxiliary minicomputer	Receiving routine Exec for auxiliary computer Character generation subroutines	3.2.2.4
	Record/replay	Record/replay mag. tape control I/O subroutine Data buffers, control	None
Executive	Real-time simulation control	Executive	3.2.2.1, 3.2.3.1.2
Library	Support	Math libraries I/O routines for peripherals	3.2.2.5, 3.2.4.7, 3.2.2.3, 3.2.4.2
Support	Ground station data	Ground station data compiler Ground station data loader	3.2.3.3, 3.2.3.3.1 3.2.3.3, 3.2.3.3.1

Table 3.1-4. Checklist for Software Requirements (Sheet 3 of 3)

Category	Function	Software Item	Applicable Paragraphs MIL-D-83468
Utilities	Prog. development modification	Resident monitor	3.2.4.1
		Assembler	3.2.4.4, 3.3.2.1.1
		Compiler	3.2.4.5, 3.3.2.1.2
		Link-loader	3.2.4.3
		Source edit	3.3.2.2
		File copy, merge, del.	3.2.4.8
		Debug (trace)	3.2.4.9, 3.3.2.1.3
		Memory dump	3.2.4.6
Maintenance	Diagnostics, calibration	CPU, mem. diagnostic	3.2.5.1
		Peripheral diagnostic	3.2.5.1
		Visual system diagnostic	3.2.5.2.4, 3.2.6
		Motion system diagnostic	3.2.5.2.4, 3.2.6
		Interface diagnostic	3.2.5.2.1, 3.2.5.2.2, 3.2.5.2.3, 3.2.6

Table 3.1-5. Checklist for Hardware Requirements

Hardware Items
COCKPIT ● CONTROLS ● DISPLAYS
MOTION BASE ● HYDR POWER SUPPLY
VISUAL SYSTEM
AUDIO SYSTEM
INSTRUCTOR CONSOLE
INTERFACE ● D/A ● A/D ● ANALOG ● LIGHT DRIVERS ● DISCRETE ● DIGITAL WORDS
COMPUTER ● CPU ● MEMORY
PERIPHERALS ● MAG TAPE ● DISC ● CR ● LP ● TPWR ● CLOCK

level of requirement definition; i.e., TS system computational system, computer program.

3.1.4.1 Criteria Categories. The criteria for evaluating the candidate training systems can be divided into two categories:

a. Training Suitability - To what extent does the candidate system configuration incorporate features which satisfy basic concepts of efficient learning.

b. Support Requirements - To what extent does the candidate system configuration minimize requirements for unique equipment, personnel, and facilities.

3.1.4.1.1 Training Suitability. The criteria for evaluating each candidate system as to the extent that it incorporates features which satisfy requirements for efficient training/simulation can be divided into eight classes.

a. Feedback - This class of suitability criteria is concerned with the extent to which the candidate system provides timely information to the student as to whether or not his response to a specific stimulus was correct. Corrective or re-enforcive information may be included in this feedback such that the student learns from his errors and his successes.

b. Participation - This class of suitability criteria is concerned with the extent to which the candidate system provides opportunities for the student to engage in practice exercises throughout the training cycle.

c. Realism - This class of suitability criteria is concerned with the extent to which the candidate system is judged to provide:

(1) The level of realism required for training of skills and knowledge for each task or subtask, and

(2) The level of realism required for valid testing of the student's ability to perform the defined job tasks.

d. Self-Pacing - This class of suitability criteria is concerned with the extent to which the candidate system permits the student to proceed through multiple training exercises at his own pace.

e. Safety - This class of suitability criteria is concerned with the extent to which the candidate system configuration reduces the potential of harm to student, instructors, and equipment with respect to actual job conditions.

f. Response Recording - This class of suitability criteria is concerned with the extent to which the candidate system provides a record of student responses to training stimuli. The importance of this factor is that it provides the instructor with a continuous basis for diagnostic of student deficiencies and planning of remedial instruction.

g. Availability - This class of suitability criteria is concerned with the extent to which the candidate system configuration reduces down-line through case of maintenance and resistance to damage by student use.

h. Flexibility - This class of suitability criteria is concerned with the extent to which the candidate system configuration lends itself to operating demonstrations, student practice sessions, simultaneous use by students engaged in independent training exercises, signal tracking demonstrations, and other instructional uses. Also included is the flexibility for updating of training sequences in accordance with mission system equipment and T.O. revisions.

3.1.4.1.2 Support Requirements Criteria Classes. The criteria for evaluating each candidate training system as to the

extent it minimizes requirements for unique equipment, facilities and personnel are divided into five classes as covered in the following paragraphs.

a. Support Equipment - To what extent does the candidate system configuration minimize requirements for trainer unique support equipment such as special test sets handling equipment and additional computer support equipment?

b. Facilities - To what extent does the candidate system configuration minimize the requirements for special facilities and services such as special structures, environmental conditioning, Radio Frequency Interference (RFI) - proofing, and electrical power.

c. Maintenance - To what extent does the candidate system configuration minimize system maintenance requirements with respect to number and qualifications of personnel, number and type of spares required, and maintenance flow times?

d. Computer Programs - To what extent does the candidate system configuration minimize the requirements for developing and support of unique computer programs?

e. New Hardware - To what extent does the candidate system configuration minimize the requirements for developing unique equipment to be incorporated into the training system?

Next, weighting factors for the eight classes of training system suitability criteria are selected on a scale of 1 to 3 on the basis of a review of AFP 50-58, (Handbook for Designers of Instructional Systems).

Training Suitability Significance (For Accomplishing Efficient Training)	Weighting Factor
Low	1
Moderate	2
High	3

3.1.4.2 Training System Ratings and Decision Table. Each candidate system is rated on a scale of 1 to 5 as to how well it satisfies each class of evaluation criteria. The rating scale for this evaluation is as follows:

TRAINING SYSTEM EVALUATION	RATING
a. Little or no capability	1
b. Satisfies criteria partially	2
c. Satisfies most aspects of criteria to a satisfactory level	3
d. Satisfies all criteria to an acceptable level	4
e. Satisfies all criteria exceptionally well	5

The product of the Weighting Factor and the Candidate System Ratings are computed for each candidate and then summed for each criteria category.

The rating figures provide a ranking of candidates system capabilities within each category and provide visibility in comparing the relative capabilities of each candidate with respect to the two criteria categories.

3.1.5 TS Software Preliminary Design

Software design requirements for TS stem from:

a. Specification of TS functional requirements (RFP spec),

b. Definition of software roles in TS operations, and

c. Descriptions of simulation events and processes to be performed or supported by software.

Training simulation software will normally be involved in the processing/presentation of sensory stimuli and in the processing/implementation of system response to student actions. Software

can also serve in executive functions controlling simulation activity and data processing/presentation of training performance.

Unlike the requirements for the TS system and the computation subsystem, detailed computer program requirements are derived by trade-offs and analyses conducted by a contractor during the proposal period (Figure 3.1.4). System and computational system requirements are derived by the Air Force and documented in the TS RFP system specification. The recipients of the RFP system must conduct trade-offs to determine which TS function should be allocated to hardware and which to computer program before computer program requirements can be derived. These and all trade-offs must be evaluated in light of the basic trade-off criteria of cost, feasibility, risk and state-of-the-art. Typically, functions which require the repetitive solution of a fixed relationship are assigned to the special purpose processor. An example of such a function is the equations which simulate the flight motion/responses of an aircraft. Processing which is not effectively done with the special purpose processor, is assigned to the general purpose computational combination of hardware and software.

When these trades are completed, detailed software requirements can be determined by the contractor and translated into his preliminary design, which is included in his technical proposal. These requirements are manifested in the identification of computer program modules and a description of the function they perform.

Several examples of analyses conducted at this level are discussed below.

3.1.5.1 Hardware-Software Trade. This example assumes a flight simulator requiring a realistic force-feel at the controls corresponding to the flight condition. The subject of the trade is whether the control loading should be accomplished entirely by software or aided by some hardware.

If all the simulated control surface positions and control-loading values are to be computed in the digital computer as shown in Figure 3.2-12, it will cost a fraction of a millisecond of each simulator cycle. The force-feed hardware will receive its commands following a large number of digital-to-analog conversions. With some smoothing circuitry added to the controlling amplifiers, the controls will not feel "steppy" with changing surface angle. If, however, the flight control surface angles are computed with analog computer components, the figure appears as in Figure 3.1-13.

Using this technique, a relatively few words of information from the digital computer suffice to drive the analog components. The trade in this case is that for the additional hardware cost of analog components, the computation time for control surface angles is saved.

Another example of a hardware-software trade is the cost of additional computing hardware versus the cost of compensating software. It is mentioned here to highlight the fact that dollar costs for off-the-shelf software and hardware are usually a minor part of the total system cost. If, for example, timing and sizing studies show that limits are being reached or the computer under consideration, buying a faster, more expensive CPU and additional memory is a relatively cheap solution compared to conversion to a lower-level language, packing data, employing sophisticated overlay schemes and so on.

3.1.5.2 Malfunction Insertion/Deletion. A software program can be provided (with appropriate interfaces to other simulator programs such as data base, executive, etc.) to allow the instructor to insert and delete malfunctions from a predetermined set. If an instructor display is available, the instructor can

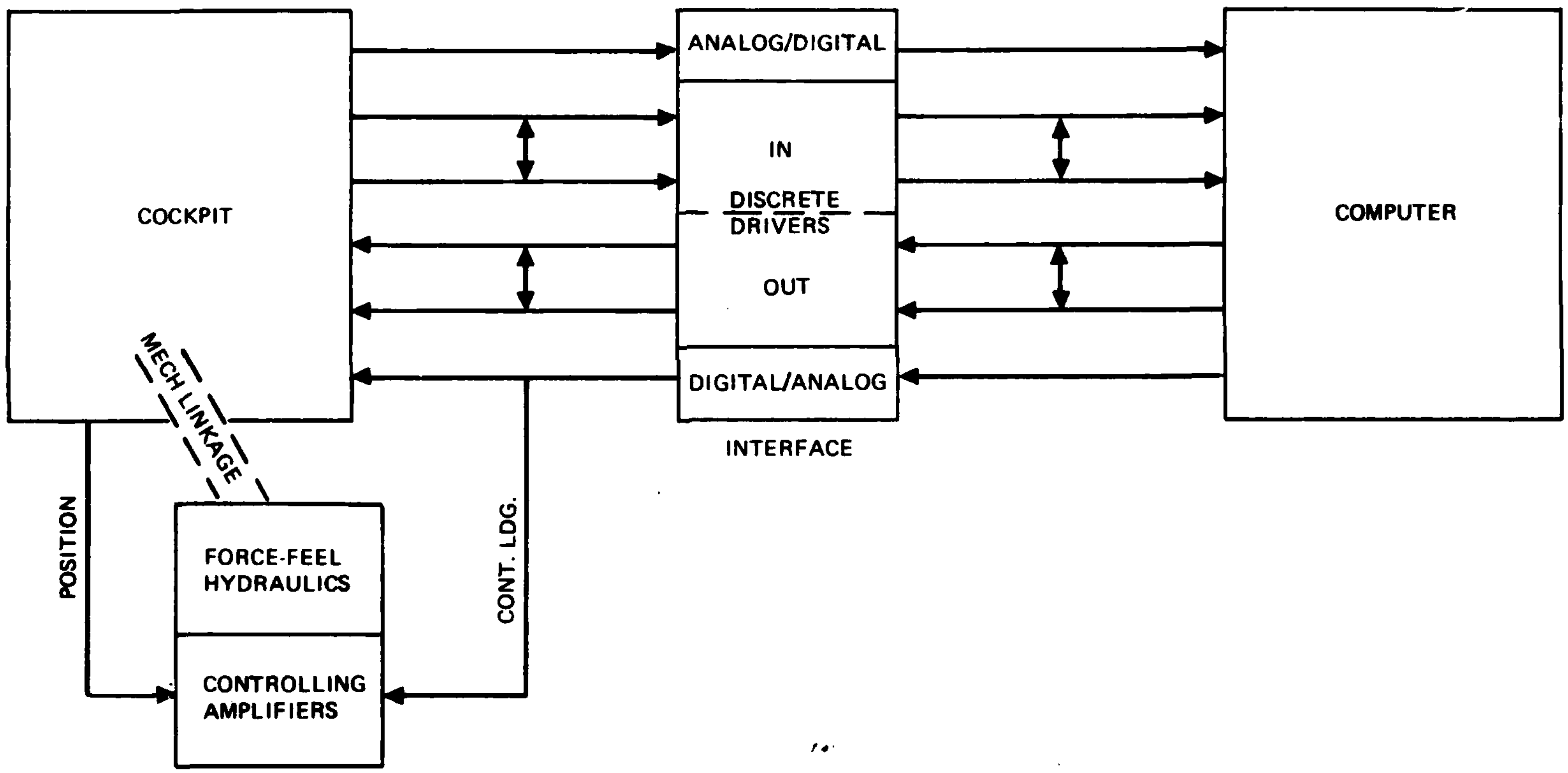


Figure 3.1-12. Flight Simulator; All-Digital Control Loading

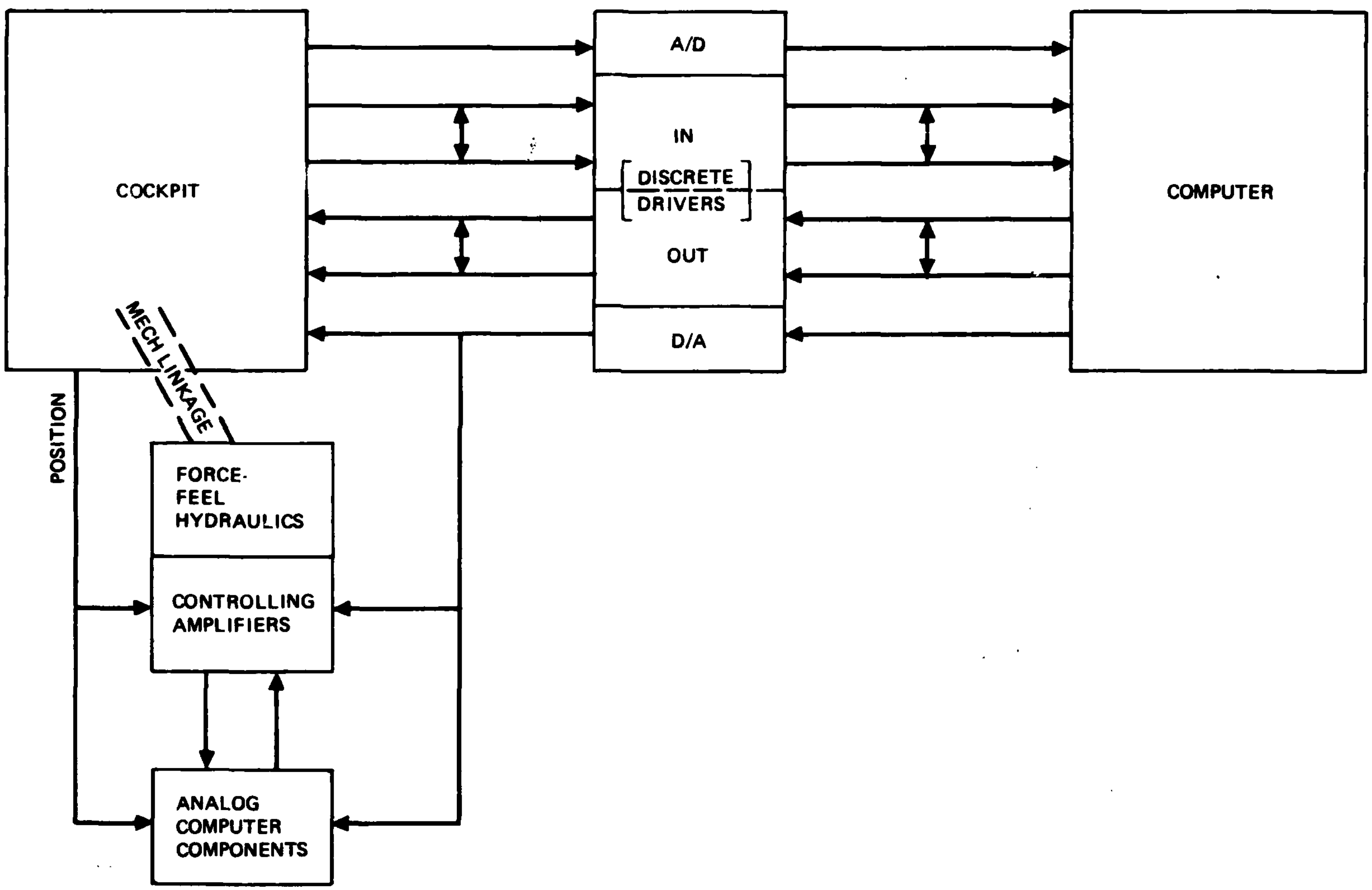


Figure 3.1-13. Flight Simulator; Analog-Computed Surface Angles

display the current status of malfunctions in effect. These malfunctions can be inserted in real-time or flagged to occur at present times. Typical malfunctions occurring in flight systems are loss of hydraulics, electric failures, etc.

A desired feature is to provide an interface with the simulator executive routine allowing reset and return to initial conditions of normal functioning.

3.1.5.3 Record/Replay. A record/replay capability may be specified in the ROC. Such a program will record on disc or magnetic tape, at basic cycle intervals, all the contents of the data base, including:

a. The state vector completely describing airplane status

b. Control inputs from the cockpit; flight controls and all pilot and flight engineer switches.

When playback is desired, through instructor request, record/replay will rewind the tape to the time desired and read into buffers the tape contents. By means of a logical "switch" in the software, all model control inputs can be taken from the recorded values in the buffers instead of from the cockpit. The airlane status will have been initialized from the beginning point of the interval being played back. A maximum time of recorded history can be specified; e.g., 15 minutes.

3.1.5.4 Display CRT. A CRT display for the instructor station can be specified in the ROC. It may be determined that an auxiliary computer is needed for this capability, to be driven by the simulation computer. The simulation computer can transmit information for display to the auxiliary computer which then generates the display symbols (alphanumeric, graphics) and refreshes the CRT image.

The essential software elements for the simulation computer are:

a. A compiler for fixed page data. These would be variable names, text, borders and title blocks. The compiler would run off-line.

b. A run-time program to fetch fixed page-data files from the disc upon instructor page request.

c. An update routine to retrieve the current values of variables and flags from the data base in real-time and combine with the fixed page-data.

d. A program to transmit the data for display through a coupler to the CRT mini-computer.

e. A program to read input from the instructor's keyboard and/or switches requesting pages and malfunction control.

The essential software elements for an auxiliary mini-computer are:

a. Application

(1) A run-time executive for program cyclic control

(2) Subroutines to generate the alphanumeric characters and graphics for display

(3) A routine to decode input character strings and call subroutines

(4) Clocked image-refreshing program

b. Minicomputer System

(1) Assembler

(2) Loaders, bootstraps and relocatable

(3) Source edit routine

(4) Debug package

(5) Dump to printer program

(6) At minimum, a paper tape system with appropriate I/O packages, interrupt handlers.

The minicomputer used for CRT display control can usually be exempted from requirements for floating point hardware, a FORTRAN compiler and a disc or magnetic tape operating system. In general, a paper tape system with tele-type and medium-speed printer will be adequate; however, the balance of the software requirements of MIL-D-83468 for systems software apply.

3.1.5.5 Real-Time Simulation Parameter Recording. A program can be provided to allow analog recording, on strip chart or X-Y plotter, of say, eight variables to be selected from the simulator data base. It may not be necessary that the variables be selectable by typing in the labels. Use of a debug routine or a core access box to insert the variables address in the appropriate location might be acceptable. Facility to scale the variables for plotting will be needed. The program will likely make use of the standard I/O routines associated with digital-analog conversion in the simulator. Storage on disc for off-line data analysis is another design alternative.

This recording capability will meet the dynamic test requirements in paragraph 4.3.10.1 of MIL-D-83468.

3.1.6 TS Software Trades

The contractor will perform a number of special software studies and trades to support TS software preliminary design as input to this TS proposal (Figure 3.1-4). Some of those studies which are closely allied to the TS configuration were described in the previous paragraphs. Additional trades more specific to software itself are described in the following paragraphs. Examples follow:

3.1.6.1 Simulator Status Update. This example assumes a system requirement that pages of simulator status shall be displayed by flight systems upon command on the instructor's CRT. These pages shall contain lists of control switch positions, cockpit instrument values, etc., labeled and updated in real time. Assume further that the requirements exist that the instructor be able, at run time, to delete any entry on a page and replace it with a different variable; there being no requirement for that replacement to be recorded back on a disc file as a permanent page change.

Preliminary design may indicate that real-time page modification requires very extensive programming, consuming a large amount of core and running the risk of being impractical due to complexity and cyclic time constraints. A trade could be effected in this case wherein the page changes are done off-line to simulation operation (relieving the space/time problem) and permanent record of the changes can be saved on disc files for future retrieval, making the change permanent.

3.1.6.2 Software Implementation Trade. TS users and builders have sought cost-effective means to maximize performance realism and minimize simulator LCC in the simulation of avionics flight software. The potential is particulary great where functions performed by flight software must be duplicated in the simulator.

TS requirements dictate that many flight software functions be reflected in the simulator. Simulation of controls and displays requires processing equivalent to the existing flight software if realism and response time are not to be sacrificed. A majority of the weapon delivery, defense penetration, and navigation and aircraft steering functions done by operational flight software are also applicable to the simulator. While some reduction in simulator computer loading can be achieved by simulating these functions, any hardware saving must be weighed against the high cost of developing and supporting a unique software package for the simulation. Consequently a trade study is conducted to determine whether the simulator data processor, including software, is an exact replica

of the system being simulated or whether a different computer and software suit is to be used.

The overall plan for this trade study is summarized in Figure 3.1-14. The first task is to scope the software elements involved in the trade. Flight software functions are evaluated for their relevance to training requirements, and for the effectiveness of the flight software modules in meeting the simulation requirements. The interfaces required are defined, including hardware and software. Interfaces are defined for incorporating the entire flight software in aircraft hardware or an emulator or incorporating only relevant modules within the simulator computer. Interfaces include not only real-world inputs to the flight software, but also simulator unique requirements for reset, freeze, mode switching, record/playback, malfunction simulation, initialization, and automatic scoring and monitoring. Avionics software functions not required in the simulator are deleted or interfaced to not interfere with simulator operations.

The next study phase defines the hardware and software configurations for the indicated design options. This definition includes all computer and interface hardware required, operational flight software, simulation and interfacing software required, and any special support hardware or software required to implement or support each option. This includes requirements impact on the simulation computer for accuracy and precision, flight software iteration rates, timing synchronization, and simulator program data structures. Special software includes required compilers, translators, loaders and utility and debug software to accommodate flight software.

The final study phase is devoted to assembling the trade analysis data and performing and documenting the trade. Hardware and software design and development costs are estimated. Simulation computer loading is used to apportion simulator computer costs for each approach. Hardware and interface complexity is evaluated for its effect on simulator reliability, maintainability and availability. Schedule problems and potential impact of high-risk items are identified for each option.

LCC are estimated for both hardware and software. For software LCC, a simulator change rate will be developed from projections from the system program and available simulator experience. This is particularly critical to the trade because the change rate of flight software has high leverage in driving the outcome of the trade. Hardware and software resources required to support each option are incorporated into the trade data.

The cost trades are combined with other, less tangible considerations, such as risk, to arrive at a recommendation. Requirements, features and impact of each option are tabulated.

The results of this study provides basis for selecting the most cost-effective method of simulation while maintaining the necessary degree of training realism.

3.1.6.3 Software Design Trades. This example assumes a visual display system is required. If the simulator is planned for night flight training, the question arises as to whether one of the computer-generated CRT displays would suffice. Note that in going to a CRT-type display, a peripheral minicomputer is probably needed for refresh of the CRT image. Appropriate software to interface between the simulator computer and the mini is needed. In trade for this added cost is relief from the electro-mechanical complexity of image generation using a moving television camera. The trade criteria described in paragraph 3.2.1 can be used to evaluate the two options.

3.2 PLANNING

Acquisition of TS systems requires coordination and planning between several Air Force organizations and one or more

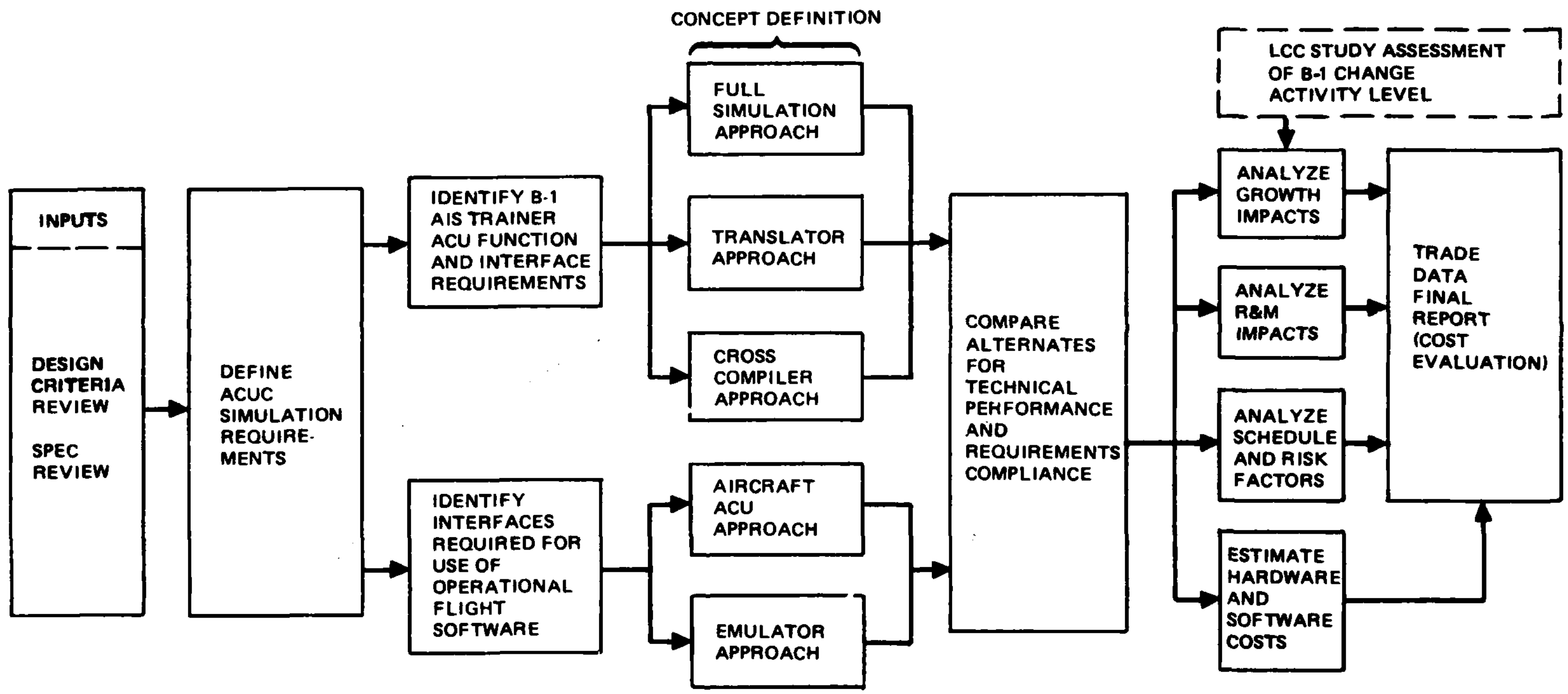


Figure 3.1-14. Computer Simulation Analysis

contractors. Planning for the software development process is contained in the following computer resource documents: (1) Program Management Plan (PMP), (2) Computer Resources Integrated Support Plan (CRISP), and (3) Computer Program Development Plan (CPDP). Specific planning for computer program requirements is not a formally documented process but is integral to the sequence of events and timing of the TS requirements specification process. The following paragraphs discuss the planning documents supporting the requirement specification process.

3.2.1 TS Software Development Planning

Planning for efficient use of computer resources is the responsibility of the Air Force. Normally, the Air Force will prepare the PMP and the CRISP, but task the CPDP to a contractor if a CPDP is required. The following paragraphs provide a brief description of the three computer resources planning documents. A more complete description is found in the Computer Program Documentation Requirements Guidebook.

3.2.1.1 Program Management. The PMP provides comprehensive planning for the acquisition management of TS computer resources. Requirements for computer resources evolve from overall system requirements via application of system engineering disciplines. Computer resources are considered as an integral part of the system and are subjected to trade-off and optimization studies along with other system elements. Refinements of these studies through system analysis result in a set of requirements (specifications) which establish in detail the required performance of each system segment and configuration item.

The PMP describes the system engineering approach to be followed in transforming operational needs into computer resources. It includes: (1) a discussion of appropriate trade-offs between hardwired digital processing equipment and programmable computers; (2) requirements for computer program and data rights consistent with the planned operational and support concepts; (3) a master schedule of major milestones, key events, and critical actions essential to timely development of computer resources; (4) requirements for acquisition and support of documentation; and (5) requirements for simulation integration and necessary support computer programs. The PMP is prepared by the Air Force in accordance with AFR 26-12 and, together with the CRISP, provides complete acquisition management and technical support of computer resources over the entire life cycle of the TS.

3.2.1.2 Computer Resources Integrated Support Plan. The CRISP identifies requirements for organizational relationships and responsibilities for the management and technical support of computer resources (as specified in AFR 800-14 Volume II). It functions during the full-scale development phase to identify computer resources necessary to support computer programs after transfer of program management responsibility and system turnover. It continues to function after this transfer as the basic agreement between the supporting and using commands for management and support of computer resources.

The CRISP is written as a part of and in parallel with the PMP. The CRISP is prepared by a Computer Resources Working Group (CRWG). The CRWG consists of representatives of the implementing, supporting and using commands to ensure that necessary elements of the CRISP are included in transfer and turnover agreements. The CRISP and its periodic updates are the responsibility of the program manager and must be approved by him. The CRISP is developed during the conceptual phase of TS system acquisition (prior to the RFP) and remains a viable document throughout the TS system life cycle. The CRISP is updated as necessary to reflect changes in computer resource requirements.

3.2.1.3 Computer Program Development Plan. The CPDP is usually prepared by a contractor for the developing Air Force

agency and is commonly required for a proposal. The CPDP may be a contractual document that applies to analysis, design, coding and checkout, test and integration, and installation (if the contractor is also responsible for the installation of the software).

The CPDP defines the contractor's overall plan for developing the computer programs and necessary supporting resources. The plan includes (1) identification of the computer program products to be delivered; (2) the development schedule and related documentation; (3) a description of the contractor development organization; (4) responsibilities for design, implementation, testing and integration; (5) hardware and facilities required; (6) procedures for managing and controlling all aspects of development; (7) a definition of the contractor's control procedures for managing design changes prior to the establishment of configuration management baselines; (8) the reporting and management of discrepancies discovered in testing; (9) responsibilities for failure analysis and correction; and (10) retesting and control of both sources and object code. If the CPDP becomes a contractual document, it would then commit contractor planning in development and control procedures for TS computer programs. The relationship of the PMP, CRISP, and CPDP planning documents to the process of deriving TS requirements was described in the previous paragraph (3.1 - Technical Evaluation).

3.3 REQUIREMENTS SPECIFICATION

As noted previously in Section 3.0, the end product of requirements specification is the procurement document called Training Simulator Requirements Specification, or simply "TS Specification." Actual preparation of the TS Specification is the subject of this section and the preparation is described with reference to technical evaluation (paragraph 3-1) and planning (paragraph 3.2).

Also noted previously in Section 3.0 is that the principal task of the AF TS software engineer in requirements specification is "to interpret and augment MIL-D-83468 for the specific TS system being developed." This concept is further expanded in the following paragraphs.

3.3.1 Stages of TS Specification Preparation

Referring to Figure 3.1-4 (paragraph 3.1), there are three principal stages to TS specification preparation:

a. Preliminary TS Specification

b. TS RFP Specification, and

c. Approved TS Requirements Specification.

The first stage is preparation of a draft TS specification based upon (1) TS preliminary design and (2) Definition/selection of candidate system (see Figure 3.1-4). Also involved in this stage (but not shown in Figure 3.1-4) is interpretation of MIL-D-83468. The checklist in Table 3.1-4 can be used conveniently at this stage, but an extensive item-by-item evaluation will be employed in the next two stages.

The second stage is a refinement of the preliminary TS specification to be included in the TS procurement RFP. Final refinement occurs in the third stage when the contractor's proposal has been submitted and agreement is reached between the Air Force and TS contractor on each item of the requirement specification.

Supporting documentation associated with the three stages is shown in Figure 3.1-5.

3.3.2 TS Specification Preparation

The TS specification contains the requirements for all the elements of the

trainer system, including those for the hardware and computer programs which comprise the general purpose digital computational system. The trainer system specification may follow the format of a Type B1 prime item development specification as described in MIL-STD-490, Military Standard Specification Practices. This B1 specification is applicable to complex items like aircraft, missiles, and "training equipment." MIL-STD-490 states that this type of specification must describe effectively the detailed performance that the item is to achieve.

The first step in preparing the preliminary TS specification (1st stage) is to determine what level specification is needed, that is, system, hardware or software. It is possible that more than one level is used. In the case of a new weapon system or a new TS system (where either complete details of the TS will probably not be known or a standard system has been selected), a system specification is appropriated. By the use of the TS characteristics checklists, the specification can be prepared. The hardware (Table 3.1-5) and software (Table 3.1-4) checklists are provided in paragraph 3.1.3.

MIL Spec (MIL-D-83468) is referenced in the proposed specification. Unique features of the subject TS can be specified by detailed description or stating deviation/limitation to particular paragraphs of MIL-D- 83468.

The functions to be performed by TS are stated first without regard to their implementation, i.e., hardware or software. This includes the weapon system functions to be simulated and establishes the required performance tolerances. It also specifies requirements for a Training Director's console and the functions to be performed at that console including any recording and playback capabilities. Computer programs are required to support the implementation of these requirements, but detailed requirements for computer programs cannot be specified at that time. The ROC provides the direct requirement that must be satisfied by the system specification. The design data package (DDP) provides supplementary data regarding the characteristics of the weapon system to be simulated. The ROC is provided by an Air Force using command and is approved by HQ USAF. The DDP is provided by the weapon system contractor. It is either included in the weapon system Contract Data Requirements List (CDRL) or is purchased directly.

The computational system is specified by referencing MIL-D-83468 Military Specification - Digital Computational System for Real Time Training Simulators. It contains general requirements for the computational system equipment and the Computer Program System. Specific tailoring of this specification must be performed to match the particular TS being developed. This specification is not intended to specify detailed computer program requirements for functions to be simulated. Rather it describes the type of computer programs that are required for the TS system.

Following the draft of the specification on the selected system, a Technical Interchange (TI) meeting is held with all interested agencies participating. Both the specification and study report drafts are reviewed. Following the TI meeting, the comments approved by the SPO should be incorporated in the specification. When the specification is released in the RFP, both the TS requirements and weapon system DDP should be part of the package for the contractor review (Figure 3.1-5).

The contractor proposal is prepared in response to an RFP which contains the TS system specification discussed in a previous paragraph. Upon RFP review, the contractor may recommend some deviations to the specification. The contractor technical proposal includes a preliminary software design for the computer programs supporting the TS system (Figure 3.1-4). The preliminary design is the result of analysis and trade study described in paragraph 3.1.5 and the information obtained in the DDP

(Figure 3.1-4). The technical proposal identifies software modules, their interfaces and describes the functions performed in each module. It is an explanation of detailed computer program requirements.

Contractor proposals are evaluated by the Air Force and the TS requirements specification may be modified as a result of the bidder's proposals or as a result of contract negotiations. In its final form, i.e., the result of contract negotiations, it becomes binding on the contractor and the Air Force and along with the contractor proposal becomes the equivalent of a development specification. The TS system will be built, delivered and accepted in accordance with this specification.

When the contractor proposal is approved by the SPO and associated agencies, the specification process is completed.

3.4 PROBLEM AREAS

The single largest pitfall in trainer software development involves "add-on" capabilities negotiated after requirements have been established. Changes are inevitable and become necessary when configuration changes occur to the system being simulated. However, frequent or untimely changes can cause significant cost, schedule and configuration control difficulties. These are nearly always reflected back to the government in the form of rising costs and increased delivery flow times. These effects can be minimized by the following actions.

a. Emphasis should be placed on producing adequate, well thought out requirements specifications. This is done by identifying the requirements, all of them, and thoroughly analyzing alternatives in the manner indicated in this guidebook before the specifications are written. In this way, the number of changes can be held to that minimum consistent with real USAF requirements.

b. Untimely changes should be avoided by incorporating changes at convenient "block" incorporation points. If possible, several changes should be collected and instituted at one time rather than incorporating the several changes independently. In this way the frequency of change is minimized.

c. The contractor should be made aware of potential changes well in advance of their need dates and his advice solicited in matters concerning implementation of the change. In this way the government benefits from the contractor's ability to assist in planning cost-effective change incorporation. Effective communication with the contractor by the TS acquisition engineer should be a continuing activity throughout the design, development test and production phases of the contract.

d. Additional problems related to the specification of TS software requirements include:

(1) Unnecessary TS software design effort can result from delayed consideration of which particular MIL Spec requirements should be exempted (for a specific TS development). Exceptions to military specification should be carefully analyzed and precisely stated prior to final approval of the procurement specification.

(2) The impact of stated requirements on TS hardware/software design is often overlooked. This results in costly system designs and/or subsequent revisions in requirements. Also, there is a tendency to require exacting performance of TS so that the best possible representation of physical phenomena is attained. Such exacting requirements may not be needed to achieve the USAF required capability. Excessively high fidelity requirements are often very costly. Further they may provide no real benefit to the TS system. Each performance requirement should be scrutinized carefully before a contracting instrument is executed, committing the contractor to meet and the government to pay for these requirements.

(3) TS system requirements providing instructor displays and controls on simulators, need careful consideration and definition. For example, more than 80% of the TS software development effort required for an advanced airborne command and control system flight simulator was expended in this area. The cost factor of TS instructional subsystems is so great that particular emphasis should be placed on TS system requirements determination to provide only that minimum instructional capability consistent with USAF requirements. This effort alone can result in greater impact to TS software requirements than the combined effect of all other system requirements. Advanced development concepts such as real-time CRT instructor displays, instructor-machine conversational input output, etc., should only be specified when these are clearly required by the ROC.

(4) Experience has shown that TS procurement dictates the need for clear definition of TS test and verification requirements. A pitfall to be avoided is the confusion caused by unclear requirements for formal TS qualification testing. Particularly important is the identification of that testing activity, including specific software tests, which are to be formally monitored by the USAF. The procurement specification should not be silent on this point.

(5) A frequent pitfall is incomplete, or improperly written Data Item Descriptions (DID). This leads to contractor misinterpretation of USAF requirements and the need for unnecessary revision of data items. DIDs should be prepared in accordance with normal practice for a weapon system. Descriptions should be complete, yet concise and free of ambiguity.

3.5 CONCLUSIONS

Several major conclusions about the TS software requirements derivation process are listed below:

a. A systematic process for requirements derivation does exist and it employs specific analysis methods; trade studies; documentation; Air Force procedures, and organization responsibilities/relationships. A composite overview of the process is provided in Figure 3.1-5.

b. TS software requirements cannot be derived independently of TS hardware and the derivation activity is a team effort.

c. Cost and other development constraints will often dictate the use of off-the-shelf hardware/software modules - to be modified and integrated for a specific TS capability.

d. A principal task of the AF TS software engineer is to interpret MIL-D-83468 for a specific TS application. Primary AF emphasis is on TS functional requirements, whereas the TS contractor will conduct detailed design trade studies to derive TS software design requirements.

e. Specific problems can be identified in the requirements derivation process but specific remedies can also be postulated.

Section 4.0. ATE SOFTWARE REQUIREMENTS SPECIFICATION

Section 4.0 identifies and describes the source of ATE software requirements; describes the process for specifying ATE software requirements; and provides guidelines for authorizing and monitoring the specification of requirements for ATE software. The term ATE refers to the hardware and software used for automatic testing. The hardware includes computing equipment, test adapters and other test equipment used for stimulus generation and measurement. Software includes the basic categories of software defined in paragraph 4.1. Much of ATE software is closely associated with the test hardware and cannot be defined separately. Thus, the process for specifying ATE software begins in the analysis required for the selection of ATE hardware even though there is little in these analyses that is directly related to software. The process of defining ATE begins with the analysis of statements in a weapon system ROC and continues until the hardware is approved in the Support Equipment Recommendation Data (SERD). Following this, the process for software requirements specification begins and continues until a development specification (MIL-STD-483), or its equivalent, has been approved for each designated Computer Program Configuration Item (CPCI). This section is organized accordingly: (1) the impact of the ROC and the weapon systems RFP, (2) the beginning of ATE requirements specification in the LSA and SERD, (3) a description of the process of deriving control and support software requirements and the procurement of a test set, and finally (4) a description of the specification of test software requirements.

ATE software requirements stem basically from two sources; from requirements related to the operability of the ATE hardware; and from ATE - independent functional and diagnostic test requirements of UUT's to be tested on ATE. However, the selection of ATE itself depends on projections of required test capabilities of UUT's to be automatically tested including the quantity of UUT's to be tested per unit of time. These requirements are directly reflected in interface test adapter (ITA) requirements, ATE stimulus and measurement capability and degree of automation in testing. Therefore, ATE software requirements specification is part of the total process of identifying and approving the support equipment for a weapon system.

This paragraph is an idealization of the sequence of events that leads to ATE software specification. The sequences described are only generally true, and are presented to give a frame of reference for discussion of guidelines and to help understand problems. Events are diagrammed in Figure 4.0-1. The weapon system ROC, the weapon system specification, and statement of work contain only limited detail on ATE requirements, and contain even less detail on software requirements; thus, they are not shown in the figure. However, system deployment and overall support concepts are defined so that operational support requirements may be derived. Basically, ATE procurement (hardware and software) depends on an identification of the operational support requirements for a weapon system. The SERD is derived via contracted LSA, an activity which is usually part of engineering development. Approved SERD's are the basis for preparation of prime item development specifications for ATE, which includes ATE software, but not the software needed to test the UUT. Software for UUT's depends on ATE/ITA design, and on the performance and diagnostic test requirements, which are documented in the Test Requirement Documents (TRD). Only those requirements in TRD's related to automatic testing are of concern to the specification of UUT software.

Figure 4.0-1 illustrates the essential characteristics of ATE software specification. These characteristics are emphasized in the figure with heavy borders.

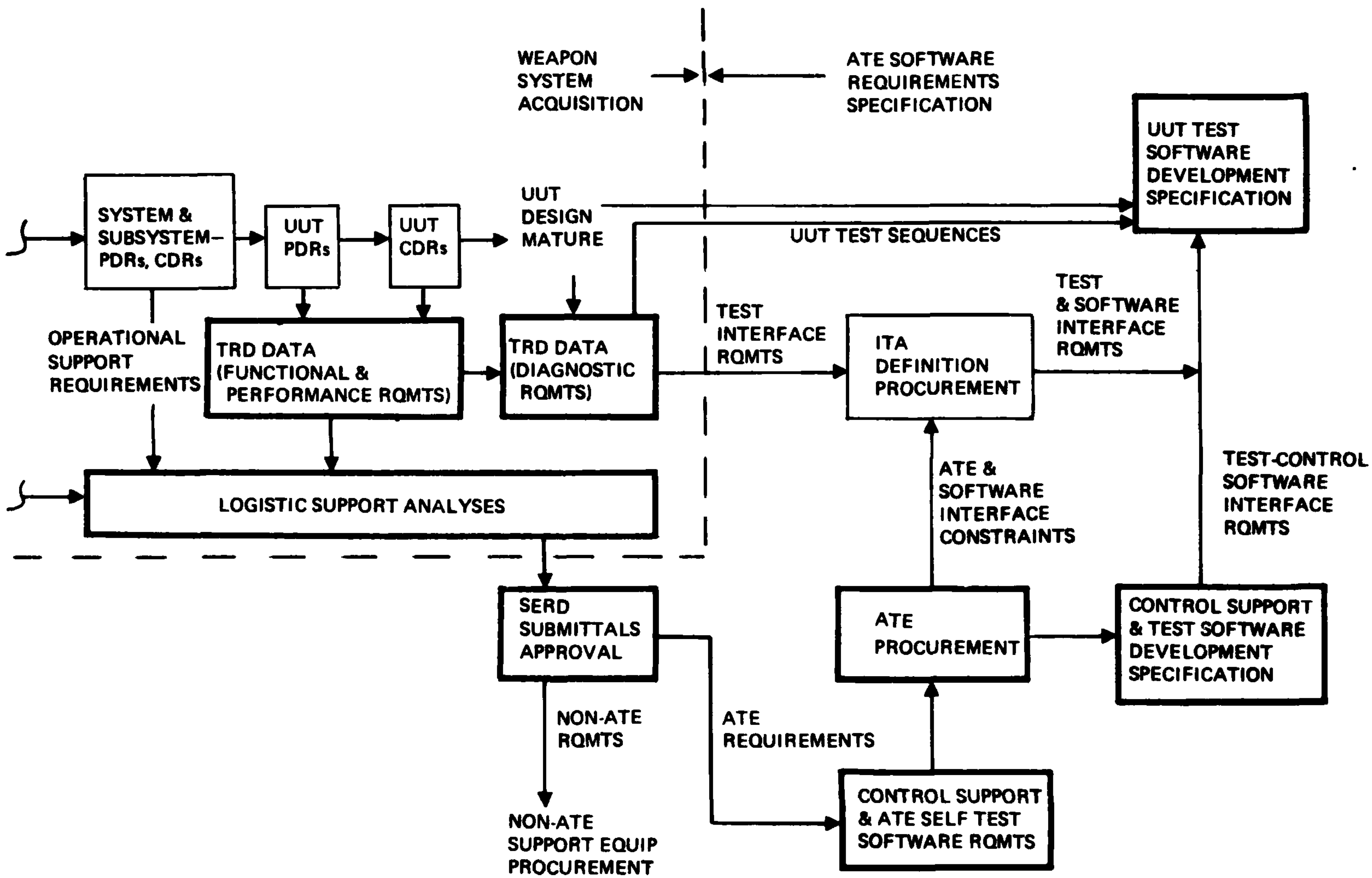


Figure 4.0-1. Key Events in ATE Software Specification

a. ATE (hardware) requirements are derived through the LSA and documented in a SERD. This process is described in paragraphs 4.4 and 4.5, respectively.

b. Control, support and self test software requirements depend on the selection of ATE and their ITA's rather than UUT design. Paragraph 4.6 describes the procurement of ATE, including the specification of control and support software (see paragraphs 4.1.1 and 4.1.2 for definitions).

c. ATE test software requirements depend on the UUT designs and are not completely defined until all UUT designs are completed and the production models are built. Performance and test requirements including test sequences are documented in TRD's. Paragraph 4.7, Test Software Requirements, contains a description of the relationship of these TRD data, the TRD and the test software development specification. The relationship of the TRD and TRD data shows how the lack of completely defined and approved test requirements impacts the development specification.

The SAE process for ATE software requirements specification begins with the requirements in the weapon system ROC and the weapon system RFP then continues as illustrated in Figure 4.0-1. It is performed primarily by a contractor with guidance provided by Air Force. The role of the Air Force ATE software engineer/manager* is to monitor the requirements specification process, provide guidance to the contracts, approve SERD's for support equipment, assist in the preparation of a contract supplement (if necessary) and approve development specifications for the computer programs to be delivered to the Air Force along with ATE.

*"ATE software manager/engineer" refers to a system project officer who is responsible to the SPO director for weapon system software, assisted at times by engineering specialists from other organizations in Aeronautical Systems Division (ASD).

4.1 ATE SOFTWARE DESCRIPTION

The three general categories of ATE software are control software, support software and test software. Each category is defined in the following paragraphs. In general, ATE control and test software operate together to accomplish UUT testing, while ATE support software assists in the development and maintenance of control and test software by providing such things as language translation capability, test station configuration management aids and program development aids.

DODD5000.29 has been interpreted to require that ATE software be treated as all DOD weapon system software; that is, be subject to configuration management per MIL-STD-483 (and other standards) and be identified as one or more CPCI's.

4.1.1 ATE Control Software

AFLC Regulation 66-37, Management of Automated Test System, provides the following definition of control software:

> "Control software is used during execution of a test program to control the non-testing operations of the ATE. This software is used to execute a test procedure but does not contain any of the stimuli or measurement parameters used in testing the unit under test (UUT). Where test software and control software are combined in one inseparable program, that program will be treated as test software, not control software."

ATE control software is designed to respond to test software to enable test functions. It also controls the ATE computer during the conduct of a test. Its source code may be a HOL such as FORTRAN IV, but often is an assembly language. Interpretive Abbreviated Test Language for All Systems (ATLAS) software systems are designed to accept ATLAS test statements directly. The interpreter makes a statement by statement translation from ATLAS to "machine" language. In this

case, the ATE control software contains a language interpretor operating on-line. For noninterpretive systems, the more usual case, the ATE control software does not contain a language interpretor and ATLAS statements must be compiled to machine language. ATLAS compilers may be executed off-line; i.e., used at a time other than testing, on the ATE computer or on a different (host) computer.

ATE control software is usually mostly made up of commercially developed software from a subcontractor. The remainder may be newly developed or modified by the prime contractor or the subcontractor.

Figure 4.1-1 provides a typical example of the composition of ATE control software. The essential functions are an operating system, a test manager, peripheral drivers, test equipment drivers, and program development programs.

4.1.1.1 Operating System Software. The operating system provides for controlling and sequencing all programs to be executed. It provides the response to all program interrupts and calls the appropriate programs in response. A test sequence will begin with the operating system initiating other control software needed to support the test and will end with the operating system ensuring that all functions are complete and accounted for.

4.1.1.2 Test Manager Software. Test manager software controls the actual sequencing of software test programs. It operates when called by the operating system software. It processes all operator interfaces and contains the interrupt processors associated with UUT testing.

4.1.1.3 Peripheral Driver Software. Peripheral driver software controls interfaces to the computer peripherals. It includes the programs to activate and deactivate the data channels and to control the flow of data to and from the peripheral devices and the computer main memory.

4.1.1.4 Test Equipment Driver Software. Test equipment driver software controls the interfaces to all test equipment similarly to the peripheral driver software for peripheral devices.

4.1.1.5 Program Development Software. Program development software provides an on-line capability for software development and the ability to make on-line modifications to test or control software. This feature may or may not be included and should be used with discretion when used, to prevent breaches in configuration management controls.

4.1.2 ATE Support Software

ATE support software consists of all auxiliary ATE software which is not normally used during the conduct of a test. Though it does not operate during the conduct of a test, it may be resident on the ATE computer. Because of planned program utilization of the ATE Station it is sometimes desirable to develop the ATE support software using a different computer. When a host computer is used (other than the ATE computer), provision must be made for the support software to execute on the host computer and generate code for the ATE computer. A compiler that is executed on a host computer and generates code for another "object" computer is called a cross compiler.

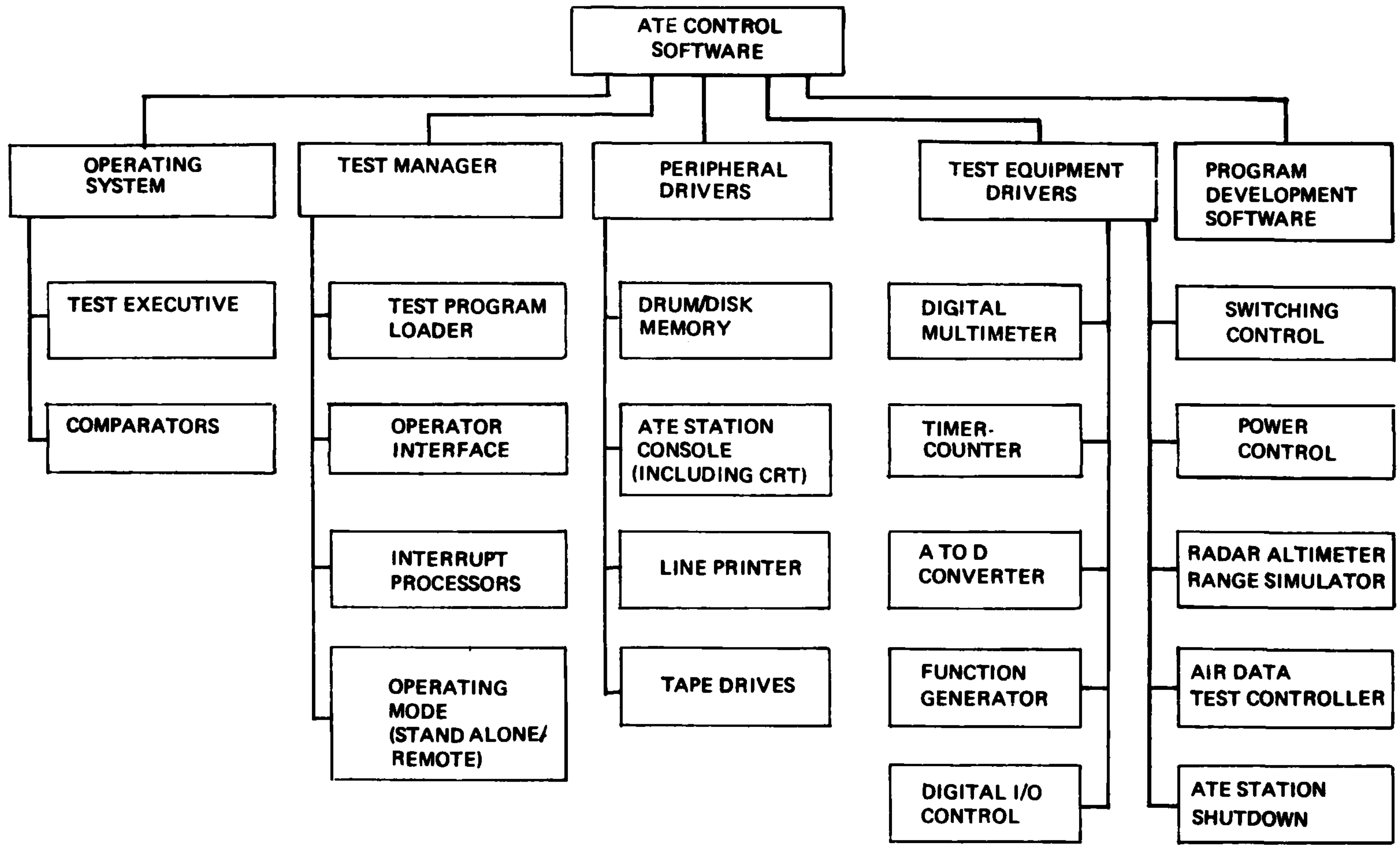


Figure 4.1-1. Typical Control Software Subsystem Structure

Language translators are required to convert the ATE source language into ATE computer executable code (machine language). A means is then required to link the code modules and assign ATE computer memory locations. Source language includes assembly language and HOL, such as Fortran IV and ATLAS. Language translators and linkers are typically a part of ATE support software. In addition to translators and linkers, ATE support software includes computer programs which can be categorized as program development aids and test station aids; e.g., a communications interface, and/or program aids, such as an automatic test pattern generator (ATPG).

ATE support software acquisition is similar to that of control software; i.e., commercially-developed software is purchased from a subcontractor which may be modified or expanded by the ATE subcontractor or weapon system prime contractor.

Figure 4.1-2 provides an example of the composition of ATE support software. The essential functions are language translators, program development aids and test station aids. The functions under these are dependent on the specific ATE application.

4.1.2.1 Language Translators. Language translators are required for all computer program source languages other than machine instructions. There must be a unique language translator for each computer and for each language. If a given computer manufacturer does not provide that language translator or it has not been developed previously, then the language processor must be developed. As stated earlier a cross compiler/assembler provides the capability for translating computer program languages for one computer on a separate host computer. This feature is usually provided by the weapon system contractor and probably requires development.

4.1.2.2 Program Development Aids. This class of programs includes all the aids for computer program development. Figure 4.1-2 shows a number of these. Some computer manufacturers have a number of highly sophisticated program development aids which can be purchased. Weapon system contractors also may have their own set of these aids. Generally the requirement for the more sophisticated aids are a function of what is available rather than a hard requirement of need. Program development aids may accelerate the coding and checkout process of computer programs.

4.1.2.3 Test Station Aids. Test station aids provide for the mechanics of joining program segments into an integral unit. It may also include computer programs for automating configuration control and computer program maintenance.

4.1.3 ATE Test Software.

Test software consists of all software used to implement documented test requirements. It consists of two types: (1) that which is unique to conducting a test on a UUT with its associated ITA, and (2) that which is used to test the ATE station; i.e., independent of a UUT/ITA. The latter test software is sometimes called "self-test" software, but is identified in this quidebook as ATE station test software. Figure 4.1-3 provides an example of the composition of test software. The essential elements are UUT test software, station test software, and ITA test software.

4.1.3.1 UUT Test Software. UUT testing is the primary test station function. A separate test program must be written for each distinct configuration of UUT. The ATE with UUT test software will be used for both performance (end-to-end/go-no-go) and diagnostic testing. Performance testing determines whether a UUT is operating correctly. If the UUT does not operate correctly, diagnostic tests are used to identify the probable failed components. The most common language used for test software is ATLAS, but BASIC and FORTRAN have been used. Test software is usually developed by the

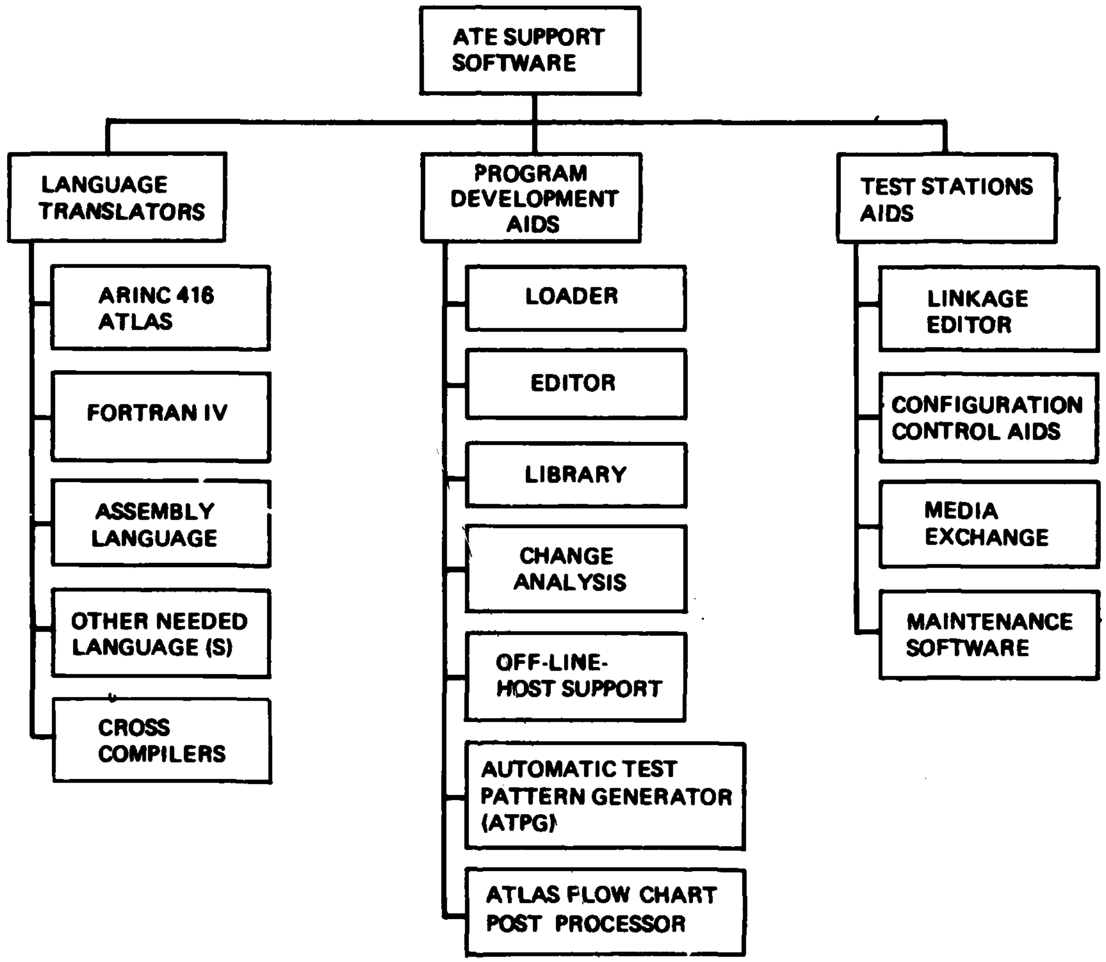


Figure 4.1-2. Typical Support Software

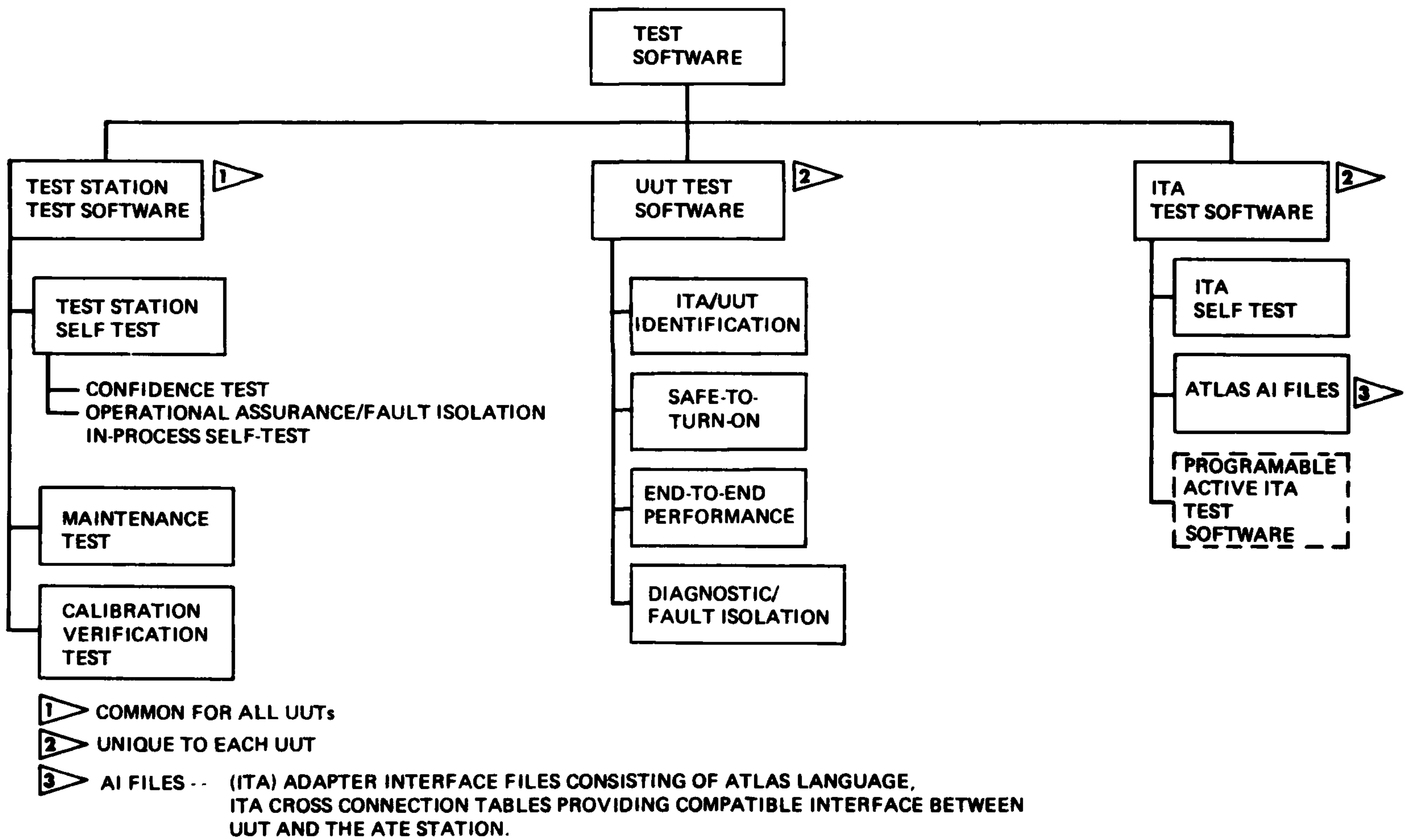


Figure 4.1-3. Typical Test Software

prime or system contractor; but if subcontractors provide some of the UUT's, they often provide the appropriate test requirements information to the prime contractor. In some cases, the UUT subcontractor may develop the test software for the prime contractor when he has the qualification to do so.

4.1.3.2 ATE Station Test Software. Station test software is used to provide confidence that the test station will perform as designed. In this case the test station is considered the UUT. Both end-to-end and diagnostic tests are performed. Station test software may be used for calibration purposes or for maintenance purposes. UUT test software is independent of the weapon system UUT. ATE station test software is usually developed using the ATE control software source language.

4.1.3.3 ITA Test Software. ITA test software is used for the same purpose as the station test software. ITA test software could be considered as part of the station test software with the exception of the dependence on the UUT. The ITA is designed to work with a UUT or set of UUT's; therefore, ITA test software is dependent on the UUT. It includes the Adapter Interface (AI) files or cross connection tables that define the interface between the UUT and the test station. ITA self test software is designed to test the ITA without the UUT being connected. ITA test software is usually written in ATLAS. If the ATE uses programmable ITA's (possibly using a microprocessor) the ITA test language will be that which is most compatible with the microprocessor selected for the ITA.

4.2 REQUIRED OPERATIONAL CAPABILITY

This paragraph and the following paragraph describe the origin of general ATE requirements. Guidelines for authorizing and monitoring the specification of requirements for ATE software depend on an understanding of the sources of requirements, constraints, interfaces, functions and quality assurance provisions. Most of these development requirements stem from the LSA, SERD and the TRD, but general ATE requirements are often found in ROC's and weapon system RFP's.

The ROC document is a formal document used to identify an operational need and to request a new or improved capability for the operating forces. This capability is described in terms of operational objectives, environment, support and maintenance concepts, and concept of system operations. The ROC is generated by a using command and approved by HQ USAF. Statements of requirements for ATE in a ROC are usually very general. Software requirements are probably not identified at all. The ATE software manager/engineer is usually not involved in either the generation or review of a weapon system ROC.

Sometimes a ROC may be issued specifically to procure an ATE capability. In this instance the ATE software manager/engineer may be invited to participate in the development of the ROC. He will then provide the technical assistance requested. An example of a ROC for ATE is a ROC issued for a Central Air Data Computer test set. Even in this case ATE software requirements are probably minimal.

4.3 WEAPON SYSTEM RFP

An RFP for a major weapon system is issued to a bidders list of prospective contractors. The elements of the RFP that are of concern for ATE software requirements specification are the Statement of Work (SOW), the Contracts Data Requirements List (CDRL) and the System Specification. The CDRL is of interest because it may specify requirements for computer program development specifications. A more thorough description of the CDRL is found in the Computer Program Documentation Requirements guidebook. Requirements for ATE could appear in the SOW and the system specification. The extent to which requirements are specified in a weapon system RFP depends on the contracting method used. ATE may be acquired by direct inclusion in the SOW and the system specifications, it

may be acquired by a contract supplement to the prime contract or it may be acquired by a separate contract. The first and third methods are not the usual, although some current weapon system contracts are using the direct inclusion and there are always isolated instances when an ATE capability is contracted separately for an existing weapon system. Requirements specification for all the methods is similar and is included in paragraph 4.6. ATE acquisition by contract supplement is the usual method and is addressed in this paragraph.

Since the weapon system RFP does not specifically address ATE requirements, the responsibilities of the ATE software manager/engineer for ATE software requirements specification are limited. The SOW must include the tasks from which ATE software requirements are conceived. They are the LSA and the generation of the SERD. These items are discussed in the two following paragraphs.

The question of whether to include ATE in weapon system RFP is the subject of trade-off. Including ATE requirements provides an emphasis on the ATE task and provides for long-lead planning. ATE has typically been de-emphasized during the early stages of a weapon system development and then received much attention when its use is imminent. On the other hand, much of the ATE and ATE software is dependent on the UUT's which are years from development. This long-time lag may invalidate the ATE requirements in the weapon system RFP and cause a considerable amount of rework and change activity. As stated earlier, both methods are being used. The trade-offs must be evaluated for the specific application. If ATE requirements are not included, the SOW should make provision for a planned Contract Change Proposal (CCP) for augmenting the prime contract at the appropriate time.

4.4 LOGISTICS SUPPORT ANALYSES

The LSA is a group of related tasks or analyses performed by the prime contractor. The objectives of the LSA are:

a. Develop requirements for support resources.

b. Assure that the support system constraints are reflected in the weapon system design.

c. Integrate the various logistics activities by maintaining a centralized source of logistics data for use by all the specialty areas in logistics.

d. Provide logistics management data to the prime contractor and Air Force logistics managers.

MIL-STD-1388-1 describes what the LSA must include. It does, however, permit the prime contractor and the SPO to implement the LSA in a manner that they feel is appropriate to the procurement.

Most prime contractors have their own worksheet formats and computer programs for summarizing the data produced. For those who do not, the Army has developed the computing software and worksheets and will provide these at no cost. The Army approach adheres closely to MIL-STD-1388-1.

A simplified representation of the LSA process is shown in Figure 4.4-1. The task analysis (Block 1) is the central portion of the LSA. It provides a breakdown of tasks required to accomplish all maintenance and general support for the equipment item. For each such task, the following data are recorded:

a. Brief description of task

b. Frequency of occurrence of the task

c. Task duration or time-to-accomplish

d. Recommended location for the task

(1) Flightline

(2) Field shop (intermediate level)

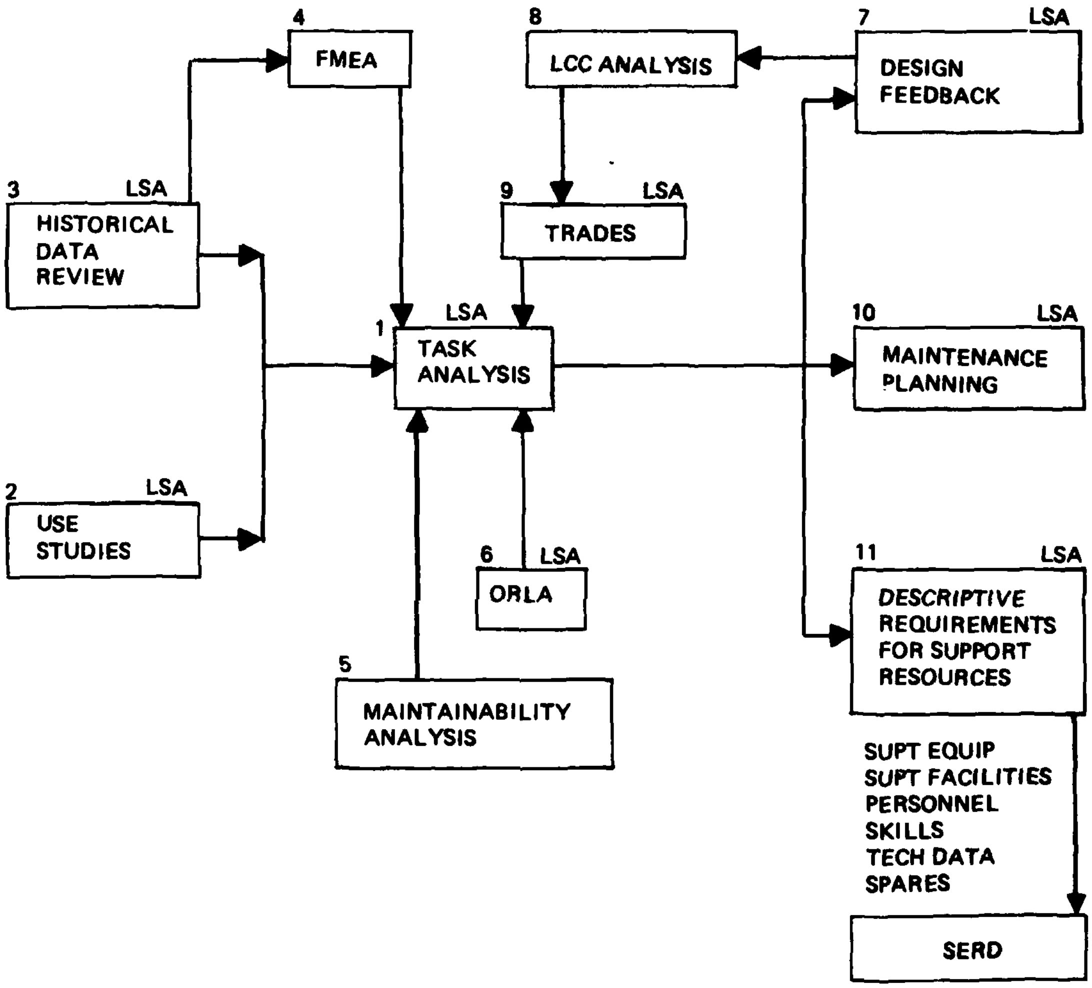


Figure 4.4-1. Logistics Support Analysis (LSA) Process

(3) Depot or contractor-furnished

e. Resources required

(1) Personnel (quantity and skills)

(2) Tools, test equipment, and other handling or support equipment

(3) Spares, repair parts, and maintenance materials

(4) Facilities

The Task Analysis is a "think-through" of the maintenance or support task by an engineer who is thoroughly familiar with maintenance of that type of equipment.

The first step is to identify the tasks to be analyzed. The kinds of tasks to be identified are classified as follows:

a. Corrective Maintenance

(1) Fault localization and isolation

(2) Remove and replace defective unit

(3) Repair defective unit

(4) Adjust or align

(5) Checkout after repair

b. Preventive Maintenance

(1) Inspect

(2) Replenish fluids

(3) Periodic replacements

c. Support General

(1) Load and unload weapons, cargo, payload

(2) Ground transportation and handling of payload

(3) Missionization of the air vehicle

(4) Towing, parking air vehicle

(5) Fueling, defueling air vehicle

Corrective maintenance tasks are identified with the aid of the contractor's failure modes and effects analysis (FMEA) - Block 4, Figure 4.4-1. Corresponding to each failure mode there are one or more corrective maintenance tasks. Preventive maintenance tasks are identified by other entries on a FMEA worksheet, such as "Life" and "mean-time-between-overhauls" (MTBO). The prime contractor's reliability program develops the data for FMEA. The support general tasks to be analyzed are identified with the aid of the use studies (Block 2, Figure 4.4-1). This LSA task is concerned with the way the Air Force intends to use the weapon system, the concept for use or employment concept. The outputs of the use studies are (1) expanded functional flow diagrams of ground operations and maintenance activities, and (2) support planning factors. Support planning factors include the following data about support general and mission elements planned for the air vehicle:

a. Frequencies of occurrence

b. Durations

c. Locations

d. Contingency operations

The third LSA task is a historical data review (Block 3, Figure 4.4-1). The prime contractor obtains experience data from the Air Force on similar air vehicles that will help him identify elements of the equipment that:

a. Are high failure-rate items

b. Are downtime contributors

c. Present safety problems

d. Are support cost drivers

e. Present gross requirements for support resources

The prime contractor's maintainability program (Block 5, Figure 4.4-1) will provide estimates of the amount of time required to perform most of the tasks to be analyzed. Other task duration data will result from the use studies (Block 2, Figure 4.4-1) and the historical data review (Block 3, Figure 4.4-1).

Maintenance task frequencies will result from the prime contractor's reliability program (Block 4, Figure 4.4-1) and the historical data review (Block 3, Figure 4.4-1). Support-general task frequencies are contained in the support planning factors produced by the use studies (Block 2, Figure 4.4-1).

An optimum repair level analysis (ORLA) is an examination of an equipment item to establish the best location for repairing it when it fails (Block 6, Figure 4.4-1). The alternatives are:

a. Discard the item on failure

b. Repair the item at the field shop or intermediate level

c. Repair the item at the depot level

All of the failure modes are examined, considering economic, operational, and other constraints. ORLA reports to the SPO include recommended repair level and the criteria and rationale used in arriving at the recommendation. The procedure for performing the ORLA is described in AFLCM/AFSCM 800-4, "Optimum Repair Level Analysis (ORLA)".

Locations for performing the repair tasks are provided by the ORLA (Block 6, Figure 4.4-1). Locations for performing the support-general tasks are produced by the use studies.

As the task analysis portion of the LSA proceeds, problems will usually come to the surface; e.g., need for test points that had not been planned. These are fed back to the designers so that corrective action can be taken. In some cases, the best alternative may not be obvious, requiring that a trade study be conducted. To support the trade study, the LSA team may require that the alternatives be costed by the contractor's LCC activity. One of the most widely used LCC models is the LSC model developed by the Air Force. In performing the trades, other potential gains and losses must be considered; e.g.,

a. Transportability

b. Reliability

c. Maintainability

d. Safety

e. Performance

f. Schedule

The design feedback, LCC analysis, and trades loop are shown in Figure 4.4-1 as Blocks 7, 8 and 9.

The maintenance-planning task (Block 10, Figure 4.4-1) starts with the maintenance concept and expands it into a maintenance plan. The maintenance plan forms the basis for tracking the other elements of logistics. Initially, the maintenance plan is made up of concepts, goals, and constraints. As the LSA progresses, the maintenance plan draws together the story of how, when, where and by whom the maintenance of the air vehicle will be done. Scheduled maintenance requirements should be planned using the methodology outlined in the appendix of MIL-M-5096D.

Among the most important outputs of the LSA are the descriptive requirements for support resources. They are developed during the task analysis portion of the LSA. Examples of these descriptive requirements follow:

a. Support Equipment: Test Amplifier input/output voltage, center frequency

b. Support Facility: 220 3-phase power supply

c. Technical Data: Amplifier voltage level, center frequency profile

d. Spares: Amplifier

e. Personnel, Skills: Radar Repair

It is level of detail in descriptive requirements that are of prime importance to the ATE Engineer.

4.5 SUPPORT EQUIPMENT RECOMMENDATION DATA

The role of the SERD in bridging the gap between the LSA and the ATE software specification is shown in Figure 4.5-1. Referring to that figure, Block 1 shows the task analysis portion of the LSA, as described in paragraph 4.4. Block 2 shows one type of output from the task analysis; i.e., descriptive requirements for support equipment, and, in this case, descriptive requirements for test equipment.

Each task analyzed by the LSA is associated with an item of equipment. That item of equipment; e.g., an avionics unit, to be tested will be called a UUT. Block 2 in Figure 4.5-1 shows UUT-oriented descriptive requirements for a task being described by the LSA to be measured and first estimate of expected values.

The LSA helps the contractor's support equipment activity to pull together or aggregate (Block 3) these requirements for one or more UUT's to develop a recommended aggregate solution. The summary of descriptive requirements is entered on the first part of the SERD. The recommended solution is on the second part of the SERD. Finally, the last part must contain procurement data such as prices, quantities, and location. DID DI-S-6176 describes the SERD once known as the Aerospace Ground Equipment Requirements Documentation (AGERD).

A SERD is prepared for each support equipment or ATE. When adapters are required, the SERD's are prepared for those also. In some cases, a SERD might accommodate a set of adapters (see Block 4 and 5 of Figure 4.5-1).

Flow time for Air Force approval of the SERD should be less than two months. If the flow time exceeds that, then the contractor may assume the SERD is tacitly approved (see Block 6, Figure 4.5-1). Approval of the SERD authorizes the prime contractor to proceed to develop ATE specifications (in the case of new-development ATE) or to initiate purchase (in the case of off-the-shelf ATE). Control and support software development specifications are generated and the test status is procured (Block 11, Figure 4.5-1).

The UUT-oriented descriptive test requirements (Block 2, Figure 4.5-1) are a collection of parameters and values to be tested. These tests are not sequenced. Block 7 shows the next step, sequencing the tests for the UUT. This defines the performance test or "go-path."

UUT requirements aggregated to one adapter (or adapter set) are grouped in order to develop diagnostic or fault-isolation test sequences (Block 8, Figure 4.5-1).

The UTT-oriented descriptive requirements for test (Block 2), the UUT performance test sequences (Block 7) and the diagnostic test sequence (Block 8) make up a significant part of the TRD for the UUT. TRD's are prepared in accordance with MIL-STD-1519 for all avionics items. The total set of ATE test software requirements are comprised of the TRD data (Block 9) and the ATE test software development specification (Block 10). This is a highly simplified description of the software requirement specification process. Paragraphs 4.6 and 4.7 provide a more complete description.

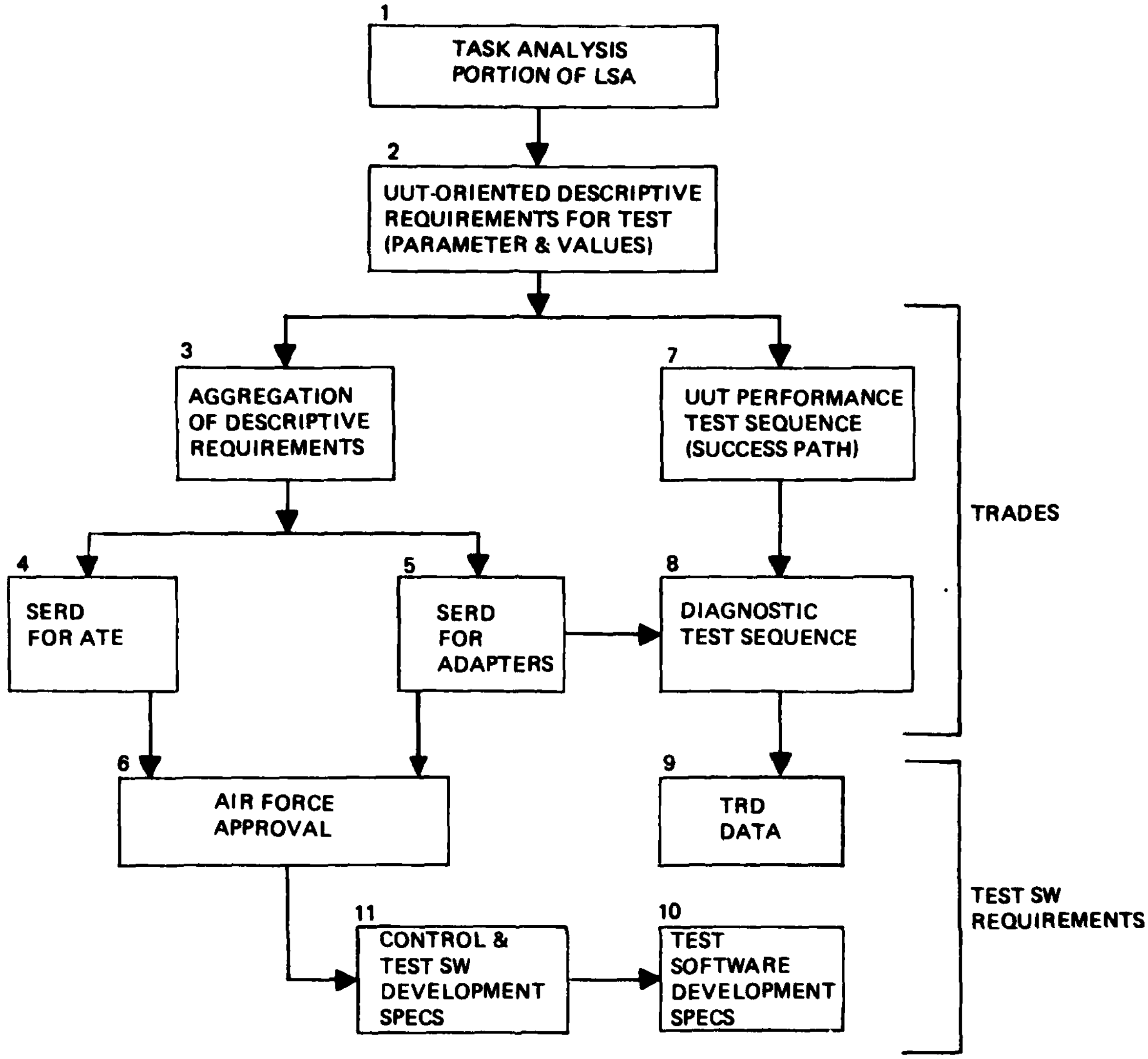


Figure 4.5-1. SERD Interfaces With LSA and ATE Software Requirements

There is an orderly methodology from the task analysis to descriptive requirements to the SERD to ATE. If ATE alternatives or options are precluded by forcing the contractor to select ATE too early, then the ATE cost-effectiveness picture is compromised.

The LSA and the generation of the SERD are contractor activities specified in the weapon system SOW and the CDRL. The Air Force ATE software manager/engineer must keep abreast of the LSA and be cognizant of the studies being performed. The primary output is the SERD which defines (for our purposes) the required ATE. Each SERD must be approved by the Air Force within a 60 day period. The SERD must be reviewed in light of the studies and reports resulting from the LSA and approved or disapproved accordingly. The approval of ATE is the real beginning for the requirements specification process for ATE software.

Usually there are only a few ATE software manager/engineers available for monitoring the LSA process. This tends to place the Air Force at a disadvantage as the contractor will employ a number of experts in the analysis. Care must be taken not to overlook this phase and to use experienced qualified personnel for monitoring the LSA process.

4.6 ATE PROCUREMENT

Procurement of ATE can begin after the SERDs are approved by the Air Force. The SERDs define the ATE that is approved for the weapon system. Computer program requirements have not yet been specified. This section will focus on the requirement specification of ATE control and support software, providing guidance for beginning ATE software requirements activity; the contracting method (a contract supplement in this case); specification of control and support software requirements, the subcontract for the test station and the role of the Air Force ATE software manager/engineer.

4.6.1 Beginning ATE Software Requirements Specification

ATE is intended to be used as operational support equipment, dealing with production equipment and as production acceptance test equipment and procedure. Therefore, as a rule of thumb, efforts for the requirements specification of ATE software should be initiated two years before the scheduled delivery of first UUT production units. The UUT source data required to specify ATE Station stimuli and sensors (and their performance requirements) and to initiate UUT test software and ITA development, would be of questionable quality if demanded too soon. Research Design Test and Evaluation (RDT&E) design evolution causes changes in the UUT designs and their associated test requirements. Also, ATE technology is developing at a rapid rate and it is desirable to take advantage of the latest technology feasible for the weapon system to be developed. The two-year flow time provides adequate time to phase ATE hardware and software requirements development.

Identification of the earliest availability of the production configuration UUT dedicated for ATE system development, determined that point during a program when a UUT has adequate design maturity. ATE station procurement can then be planned and scheduled as shown on Figure 4.6-1. "Zero" time on the chart is the start of UUT test software specification. This figure shows flow time keyed to a point at +1 years, which is the availability of the first production UUT dedicated for ATE system development. At this point software integration and software-hardware demonstration tests can be accomplished. Backing up in time from this point, ATE station procurement is shown together with normal acquisition steps.

A scheduling conflict is noted when comparing the target date for ATE, Station Stimulator/Sensor requirements data and

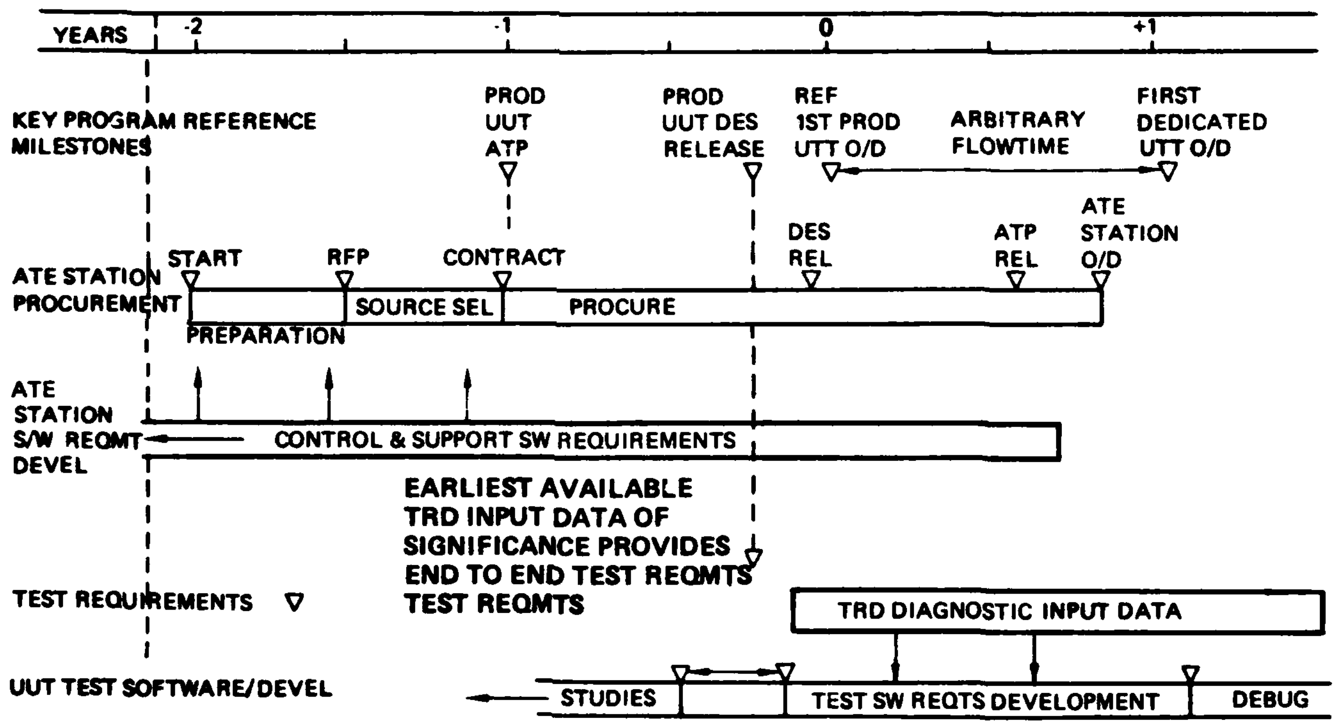


Figure 4.6-1. Phased Flow-ATE Development and ATE Software Requirements

the availability of TRD input data (particularly diagnostic data). From a standpoint of true need, these schedules should be reversed. TRD input data are needed to specify characteristics of the ATE station stimuli; sensors, ATE control software timing; and ATE station resources allocation. Despite this conflict, Figure 4.6-1 shows ATE software requirements development and procurement in two phases. The first phase addresses ATE control and support software, i.e., part of the ATE station package. The second stage will address UUT test and ITA test software.

4.6.2 Contract Supplement

Following approval of the SERDs, some method of contracting for ATE procurement is usually needed. Some form of contract supplement, such as a CCP, is commonly used. Other methods were identified in paragraph 4.3. A detailed discussion of contracting for ATE computer program is provided in the "Contracting for Software Acquisition" guidebook.

At this point the project office usually will request the weapon system contractor to prepare an amendment to the weapon system SOW for the inclusion of ATE. The number of Air Force ATE software manager/engineers assigned to a project is usually severely limited. Therefore, their participation in preparing the SOW for the CCP is monitoring the activities of the contractor and providing technical consultation in defining the tasks related to ATE software requirements specification and software development. The SOW should define the tasks of software requirements specification and specify the requirement for a computer program development specification for each CPCI. In addition, the SOW should specify the applicability of an ATLAS language specification. The AF ATE engineer should ensure that the SOW states that UUT test software and ITA Self-Test shall be done in ATLAS in accordance with an identified ATLAS language specification. The SOW should state that a HOL, preferably FORTRAN IV be used whenever possible in any newly developed control and support software and that assembly language be used only when it is impractical or impossible for a program coded in the HOL to satisfy the program requirement. Additional considerations to be included in the SOW are as follows:

4.6.2.1 Security Provisions. When the ATE software is required to process classified information, the contractor should include in his proposal the administrative, physical, and personnel security measures required to protect the classified material, together with his plans for implementing these measures.

4.6.2.2 Support Software Training Requirements. If a significant amount of ATE support software is anticipated, it may be appropriate to direct the contractor to provide an estimate of the requirements and recommended approach for training the personnel needed to develop ATE support software.

4.6.2.3 Data Rights. The Air Force, particularly AFLC, may anticipate that it will have further need of computer programs or data generated under the contract. In this situation, a statement stating that the contractor is required to give up proprietary rights to the subject computer programs is required.

4.6.2.4 Nuclear Safety. AFR 122-10 states that the software used for testing or checkout of aircraft or missile systems employing nuclear weapons must meet certain safety criteria. Specifically, 122-10 states that any software which can exercise automated or automatic control over any critical nuclear weapon system function must be subjected to a software Nuclear Safety Cross-Check Analysis (NSCCA) to ensure system nuclear safety integrity. Critical functions are those which apply directly to, or control, the prearm, arm, fire, unlock, release, launch or targeting functions of a nuclear weapon system. The contractor should be instructed to discuss the aspects of these as it applies to the ATE system.

4.6.2.5 Software Design Approach. Organic software maintenance considerations may dictate that HOL be used wherever reasonable because the use of HOL normally makes that maintenance more efficient. Also, where new software is to be built, the contractor will often be directed to use a modular design approach such as the top-down structured approach. General statements are often used to call for software design which is consistent, logical, and well documented in accordance with stated standards.

4.6.2.6 Growth. One of the factors to be considered in ATE design is the growth capability required by the ATE computational system. Growth potential must accommodate the predicted level of computer program change and growth activity over the anticipated life of the system. The contractor should be directed to estimate the required growth features such as spare central processing time, spare memory, and spare input/output channel capacity and provide a specified quantity of growth capability.

Following the preparation and submittal of the CCP, the Air Force ATE software manager/engineer will participate in the review and approval cycle. Following approval of the CCP, work can begin on the process of computer program requirement specification.

4.6.3 Control and Support Software Requirements

Much of the control and support software required for an automatic test station can be purchased from the ATE computer manufacturer. Many times, the control and support software requirements are "defined" by studying the off-the-shelf programs in a particular computer manufacturer's inventory and specifying what is available. Some systems will work acceptably in the manner, thus this technique can be used to some extent. A more acceptable approach is to define the requirements for the specific test system being designed, then include these requirements in an RFP to prospective test station subcontractors. This approach may require the subcontractor or the prime contractor to develop some computer programs to satisfy this requirement. These requirements for control and support software are derived by the weapon system contractor and recorded in the prime item specification to be included in the RFP for the prospective test station contractors.

4.6.3.1 Control Software Requirements. The purpose of ATE control software is to provide a workable test system, providing interface between the test operator, test software and the test equipment including the ATE computer. Consequently, ATE control software requirements cannot be derived independently from the test equipment and in fact must be derived in parallel with the equipment. Following the LSA and approval of the SERDs the efforts required for specification of control software requirement are as follows:

First, a refinement of the general concept of the ATE station, used in developing SERDs, must be accomplished. Second, in parallel but slightly lagging SERD development, a study must be accomplished to document the physical and functional interfaces of all UUTs to be tested. Third, an estimate must then be made of the number of stimultaneous stimuli and sensor measurements which are required to be applied by each stimuli for each UUT via the physical interfaces. The total set of UUTs are then examined and the worst case of simultaneous usage for each stimuli and sensor is determined. The overall workload of the test station is examined to determine the total number of UUTs to be tested simultaneously. Total station through-put will impact the control software requirement.

In parallel with this effort, the capability of available ATE station computers should be surveyed with emphasis on capacity and speed. Speed is essential to provide the ATE control software with adequate timing characteristics. (The timing requirement generally requires

that much of the executive software be written in assembly language, though the rest of the control software might be written in an HOL). Computer speed is the deciding factor in determining whether or not such analogue functions as rise times, delay times, sequenced switching times can or should be accomplished by hardware or ATE control software.

At this point, studies should be conducted jointly by the UUT and ATE engineers trading off percent of achievable UUT maintainability versus ATE Station requirements (both hardware and software). The greater the required achievable UUT maintainability (over and above mandatory end-to-end testing) the more stringent the requirements on ATE control software in all three areas of executive software responsibility (see paragraph 4.1.1). In addition, the specification of other modules of ATE control software are affected either by a greater quantity of requirements (e.g., large amount of test equipment to be driven and more interrupt processes) or by a requirement to handle a larger volume of data. ATE station test software requirements are correspondingly greater and the quantity of UUT test software requirements result in a larger volume of ATLAS statements. The eventual result of these study efforts provides a set of fundamental (top level) requirements for ATE control software.

Prior to or in parallel with the above studies, the ATE control software must be conceptually configured to include the functions described in paragraph 4.1.1, ATE control software definition. The above determined top level requirements for simultaneous measurements are analyzed and allocated as requirements on the ATE control software, particularly on the test equipment driver and the operating system software functions.

As a result of certain LSA efforts, (paragraph 4.4) the need for displaying and providing hard copy data will have been established. From follow-on LSA human engineering studies the form, content and timing of the displayed data and hard copy data will have been determined. Both of these study results provide specific data processing requirements for the ATE control software, particularly the operating system and peripheral driver software.

The requirements for the remaining ATE control software modules can usually be selected from available specifications for commercial computers.

4.6.3.2 Support Software Requirements. ATE support software, consists of the three primary classifications discussed in paragraph 4.1.2. Computer program requirements are derived largely from high level requirements such as the use of FORTRAN and ATLAS, and the necessity for using an assembly language. Program development aids are, for the most part, standard equipment for the computer manufacturer. ATE computing equipment manufacturers have developed a variety of these aids, some more advanced and sophisticated than others and some with better track records for dependability. Figure 4.2.2 should be used as a guide for specifying the types of support software required. In many cases the weapon system contractor has his own library of support software or will specify support system requirements that require a development effort. The requirements for station and program aids result from cost versus utility studies. Once such aids are determined to be beneficial, these requirements enumerate functions and specific inputs/outputs.

There are occasions when it is desirable to develop ATE computer programs on a larger more powerful computer than the one selected for ATE. This is the subject of a trade-off. The off-line computer represents an additional expenditure for equipment that can be traded with improvement in flow times for new software development and for computer program maintenance. The improved flow time stems from a larger more powerful computer and better program development aids. Another factor is the availablity of the ATE computer if testing activity

is high and whether all program maintenance activities can be performed as a central location. The choice of a "host" computer can affect the requirement for support software, e.g., this type of operation would require a cross compiler. There must be a close relationship between the host and target computer hosted support software.

In summary the support software requirements specification process is usually an activity performed by the contractor using his experience, expertise, and knowledge of available support software. The role of the Air Force ATE software manager/engineer is mainly to monitor the process and to give advice and council during the process. Depending on the CCP SOW, the Air Force may or may not have approval rights over the prime item development specifications for the test station which includes the control and support software requirement.

4.6.4 Test Station Subcontract

The usual method for a weapon system contractor to acquire an automatic test set is to purchase the test set and the control and support software from a supplier. The supplier then becomes a subcontractor to the weapon systems contractor. This removes the Air Force ATE software manager/engineer further from the requirements specification process. The Air Force has no official jurisdiction over the subcontractor, only the prime contractor. The Air Force engineer is usually invited to attend technical reviews and is usually a recipient of all documentation produced by the subcontractor. He may provide counsel and offer suggestions but cannot provide direction except through the prime contractor. Any action that may be taken that is out of the subcontract must be negotiated with the subcontractor and the prime contractor and will probably result in a cost adjustment.

4.6.4.1 RFP Preparation. Since the Air Force has no jurisdiction over the subcontractor, it is important for the software manager/engineer to closely monitor the preparation of the RFP to be issued by the prime contractor to the test station bidders.

The control and support software requirements described in the previous section are recorded in a Prime Item Development Specification for the test station. This specification is usually written in accordance with MIL-STD-490. If the test set vendor is to be selected through competitive bidding, the requirements will be general and will not address a particular vendor's software implementation. The specification will describe what functions the software must provide and any implementation requirements which are important to software adequacy. This specification defines each of the software functional capabilities, defines how they relate to one another, how it ties into the ATE hardware (computer, peripherals, test hardware) and defines the qualification test and acceptance test requirements for the software.

In addition, the RFP should instruct the test station subcontractor to prepare a computer program development specification (as specified in MIL-STD-483) for control or support software that may be developed. These CPCI development specifications are the final step in requirement specifications for control and support software. The RFP should also instruct the bidders to provide program listings and other design description data equivalent to a CPCI product specifications (also specified in MIL-STD-483) for off-the-shelf computer programs that he will supply. (It should be noted here that any new programs or modifications to existing programs that may be developed by the prime contractor should also require a CPCI development specification.)

4.6.4.2 Guidelines for Authorizing and Monitoring ATE Control and Support Software Specification. Depending on the language of the CCP SOW, the ATE software manager/engineer may have the authority for review and approval of the test station prime item development specification and to assist in the evaluation and

award of a contract to the test station subcontractor. Assuming this is the case, the appropriate points for monitoring the specification of control and support software are related to the usual major system procurement and system development steps. In summary the ATE software manager/engineer must be involved in the following activities:

a. Determining if the specific requirements of ATE software are included in the ATE procurement package, as described in paragraph 4.6.3;

b. Approving the statements of software operability and software requirements in the ATE "A" type specification or in the ATE prime item development specification;

c. Assisting in ATE contractor evaluation by assessing the ATE contractor's system software development credibility and capability;

d. Determining if the ATE contractor is compliant with the intent of the SOW and CDRL requirements for tasks and documentation of software requirements;

e. Determining if specific attention has been given to the maintenance of control and support software; and

f. approving the development specifications.

Paragraphs 4.8 through 4.11 of AFR 800-14, which cover program technical control and review, are applicable to this part of the specification of ATE control and support software. If ATE procurement is being managed by a contractor for ASD, then these engineering management requirements should be made his obligation.

4.6.4.3 Computer Program Development Specifications. Computer program development specifications generated by the test station subcontractor will complete the requirements specification process for control and support software. These CPCI development specifications contain the detailed requirements for computer programs that must be developed. These specifications should be reviewed and approved by the prime contractor prior to the software Preliminary Design Review (PDR). Formal reviews should be conducted to evaluate and approve the specifications. The Air Force is normally invited to attend, although they have no official jurisdiction as stated earlier. Air Force representatives may comment and provide guidance and may offer direction only through the prime contractor.

4.7 TEST SOFTWARE REQUIREMENTS

As defined previously, test software consists of UUT test software, ITA test software and ATE station test software. Test software requirements depend on the selected ATE and on UUT design data, whether the UUT be the ATE station, an ITA or a Line Replaceable Unit (LRU), and on the defined set of sequenced tests which have been approved by the UUT design organization. Test software requirements specification is complete when the test sequence stimulus, measurement and ancilliary data has been approved, or an approved source (such as an approved TRD) has been referenced.

The following paragraphs define source data for each category of test software and discuss the relationship of source data to TRDs in test software requirements development, the test software requirements development process, and the test software development specification. Guidelines for the involvement of the ATE software manager/engineer are also included throughout.

4.7.1 Test Software Requirements Source Data

4.7.1.1 LRU, Secondary Replaceable Unit (SRU) as the UUT. Input data to UUT test software requirements (both end-to-end and diagnostic testing) are:

a. UUT level and UUT sublevel acceptance test procedures and associated

test requirements (including Contract End Item (CEI) Specification);

b. UUT level and UUT sublevel logic, functional block, schematic, single function and wiring diagrams;

c. Appropriate references to replaceable units of the UUT;

d. Descriptions from a qualified design engineer of the workings of the UUT circuits and methods of troubleshooting them; i.e., diagnostic testing and fault isolation. The purpose of these data are to educate the contractor's UUT test software designer on how the circuits work, the critical functions and potential failure points so that he can properly select, organize and sequence the ATLAS programming effort during CPCI development; and

e. ATPG data for UUT digital circuit functions.

4.7.1.2 ITA as the UUT. The source data for ITA self-test and diagnostic software and test software for adapters is equivalent to that described in the paragraph above with the letters ITA substituted for UUT. Subparagraph (e) applies only to programmable, active adapters.

4.7.1.3 ATE Station as the UUT. The basic requirements for ATE station test software are (1) to provide a means for assuring adequate operability of the ATE station (usually accomplished by end-to-end tests), and (2) to provide ATE diagnostic test capability. The source data to satisfy both basic requirements for ATE station test software is equivalent to that described in the paragraphs above. An ATE station acceptance test fixture, which allows more complete ATE end-to-end testing is frequently employed. Product specifications for the ATE are the source of both performance and diagnostic test sequences. ATE station test software will usually be provided by the ATE contractor, but there are examples of test sets that are both ATE and flight readiness equipment. In the latter instance, station tes sequences may be prepared by the "prime contractor with assistance from the ATI manufacturer.

4.7.2 Relationship of Test Software Source Data and Test Requirements Document (TRD)

Contractors normally separate logistics engineering from design engineering organizationally. TRDs are prepared under the cognizance of the contractor's logistic organization for the AFLC because logistics personnel deal with AFLC constantly, know their needs, problems, and how they operate.

Historically, the contractor's logistics organization has prepared TSs for end-to-end and diagnostic testing of UUTs in parallel with software development and then demonstrated their correctness by technical order verification and validation (V&V). Technical Order (T.O.) V&V has been conducted, as a "hands on" operation for participating AF personnel.

The contractor's logistic organization obtained the T.O. source data by requesting T.O. inputs from design engineering. These input requests were processed through the contractor's change board, then scheduled and documented as formal data packages to be provided by design engineering to the logistics organization.

Currently these T.O. input data packages are to be included as part of deliverable TRDs (in addition to the T.O.s). This will not, however, change the contractor's internal mechanism for developing the data packages. Design engineering will continue to develop these inputs and forward them to the logistics organization for incorporation into TRDs.

Avionics design engineers developing the UUT and ITA test software will normally be located in the contractor's design engineering organization. The major problem associated with test software development is not "were the tests written correctly in ATLAS?," rather

"were the correct tests written in ATLAS?" This problem can only be solved by comprehensive review by the design engineer and the test software engineer, using either ATLAS language or English language statements, depending on which provides the best means of communication.

The contractor's UUT and ITA test software designers will obtain their source data directly from TRD data provided by design engineering and not wait for the logistics organization to prepare and process TRDs. In some instances these source data are provided in the ATLAS language.

TRDs are only one source of data for UUT and ITA test software definitive requirements. Other sources are the many informal discussions with UUT design engineers, where specific questions are answered, circuit understanding is obtained, and design features are interpreted.

The AF ATE software manager/engineer has the opportunity to review these test sequence data at formally scheduled TRDs. These data are scheduled in a sequential series starting with requirements for end-to-end testing and continuing with sets of diagnostic testing requirements.

4.7.3 UUT Test Software Requirements Development

The end-to-end test concept and general plan can be started immediately after the UUT production acceptance test procedure is available. The diagnostic test concept and general plan is delayed due to the dependence on TRD input releases.

4.7.3.1 End-to-End Tests. The UUT end-to-end software requirements definition process is illustrated by Figure 4.7-1. Available UUT source data (see paragraph 4.7.1) for end-to-end tests are compiled and put into a form facilitating the generation of UUT test concept and general plan. The total functional capability of a UUT is broken down into a set of logical subfunctions, the proper operation of which will be verified by a set of tests; e.g., synchro operation under varying power conditions. The number of UUT test points impacts the programming test and software development cost. This leads to a trade-off between availability of test point and program complexity with cost as the primary criterion. Depending upon the complexity of the UUT it may be desirable to make one or more additional levels of functional breakdown. A UUT general test plan is generated which defines the selected sequence of subfunctions.

Upon completion of the UUT test concept and general plan, an in-house review is held with the UUT design engineer(s), to (1) assure that the correct UUT design and test requirements data were used, and (2) to establish approval of the UUT test concept and general plan.

The contractor's UUT test software engineer then lists all the detailed tests he intends to perform to exercise adequately each UUT function in a manner which will satisfy the UUT acceptance test procedure. These tests are then sequenced and flow diagrams may be prepared. Figure 4.7-2 is an example of a simple diagnostic flow diagram for certain synchro operations. The main flow portion of the diagram could be a part of an end-to-end acceptance test. The flow diagrams are then reviewed with the UUT design engineer for concurrence that these tests will satisfy the UUT acceptance test procedure or equivalent. At the present time, english language statements are preferable because many design engineers do not have working knowledge of ATLAS, although with some training this problem can be overcome, and reviews are simplified. With the current level of familiarity with ATLAS, English language flow diagrams have particular value as a tool assuring that changes in ATE stations or UUT configurations are adequately incorporated. As the ATLAS language becomes better known, it may provide a more precise method of communication than English.

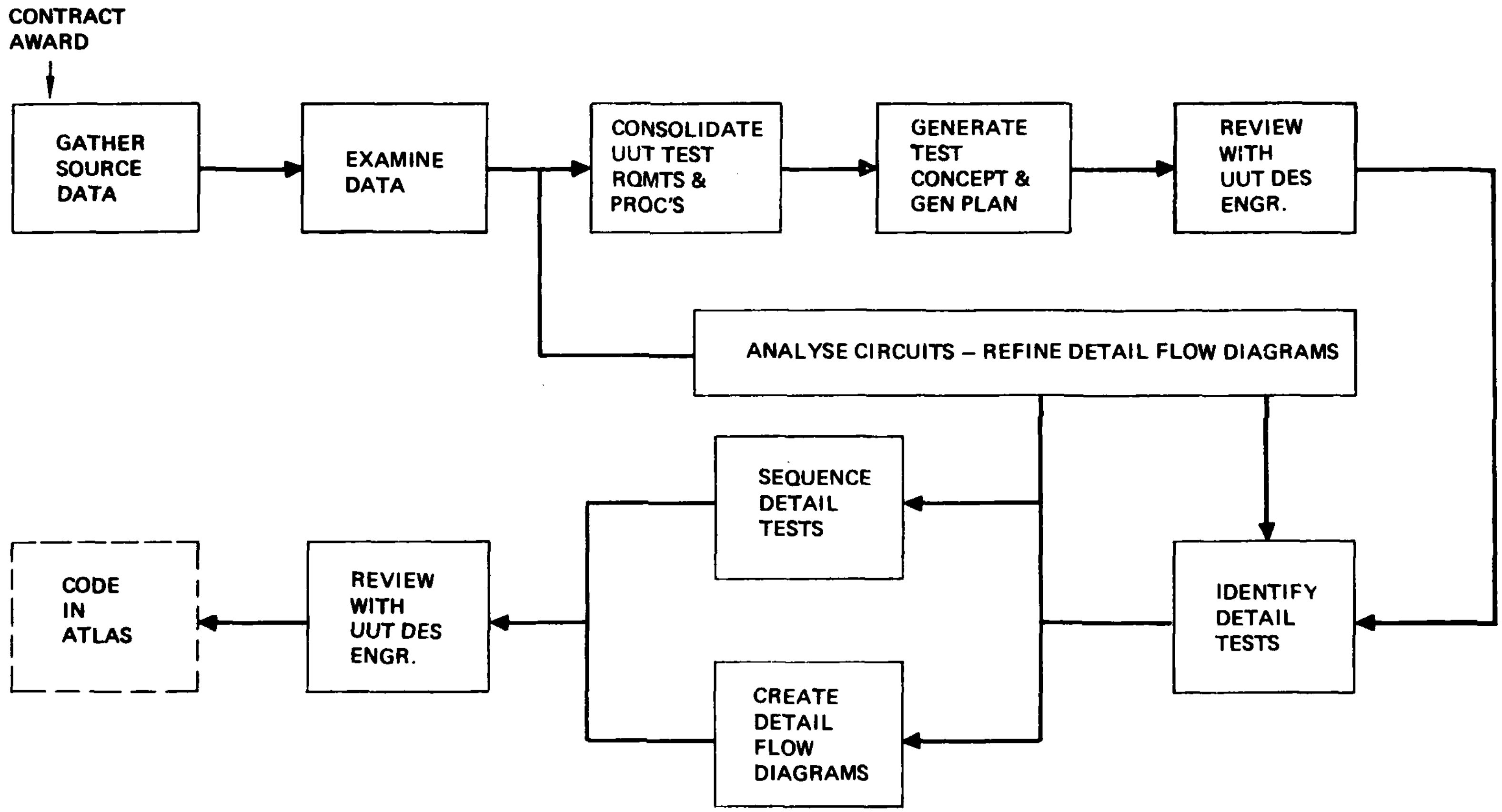
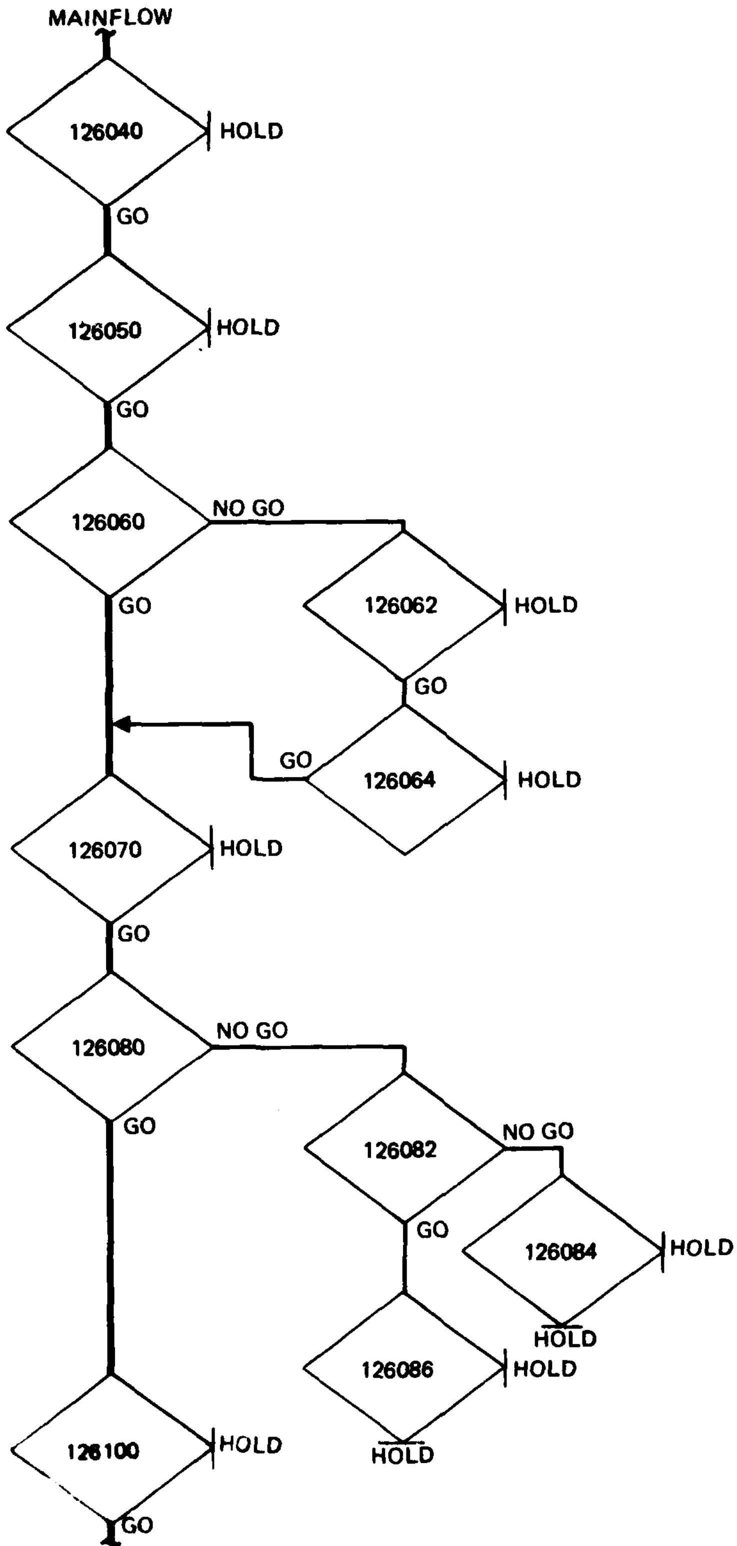


Figure 4.7-1. Example UUT End-To-End Software Requirements Definition



MEASURE SYNCRO CHANNEL A
IF HOLD REPLACE PCB 45

MEASURE SYNCRO CHANNEL B
IF HOLD REPLACE PCB 46

MEASURE SYNCRO A PEAK

MEASURE SYNCRO A VALLEY
IF HOLD REPLACE PCB 42

MEASURE SYNCRO A PEAK
IF HOLD REPLACE PCB 41

MEASURE SYNCRO SIGN BIT
IF HOLD REPLACE PCB 32

WITH SIGN BIT 'HIGH' MEASURE
SYNCRO B PEAK

MEASURE SYNCRO B VALLEY

MEASURE SIGN BIT INPUT
SIGNAL 'LO [illegible] 'GO HOLD'
REPL ACB 21
IF 'NO GO HOLD' REPL PCB 22

MEASURE SYNCRO A FOR
CROSS-TALK
IF 'GO HOLD' REPL PCB 22
IF 'NO GO HOLD' REPL PCB 18

MEASURE SYNCRO SIGN BIT
IF HOLD REPLACE PCB 22

Figure 4.7-2. Typical Flow Diagram

The form in which the UUT test software requirements is presented is the subject of a trade-off. There are two opposing views. One is that TRD data should be written in English by the UUT design engineer and converted to ATLAS by a test software engineer. The second is that the TRD data should be written directly in ATLAS by the UUT design engineer. Arguments for the first case are alluded to in the above discussion. The argument for the second case is that there may be an information loss in translating the UUT designer inputs to the flow charts and into ATLAS and that a review cycle could be eliminated. It is currently true that most design engineers are not familiar with ATLAS, but as stated previously this problem can be overcome with adequate training. If the UUT test sequences are preferred in ATLAS, the test software engineer should provide appropriate guidelines for the UUT design engineer to follow when writing ATLAS statements.

Certain groupings of these test procedures will be identified as CPCIs. For each CPCI certain configuration management functions must be performed such as preparation of development specifications and product specifications and holding preliminary and critical design reviews (PDRs and CDRs). The design reviews provide an opportunity for the Air Force to participate in the review process and to offer guidance and direction as necessary. The test software development specification is described in paragraph 4.7.6. The test sequence flow charts and the ATLAS statements are included in the product specification and are reviewed at the CDR for the specific CPCI.

4.7.3.2 Diagnostic Tests. The process for development of UUT diagnostic test software requirements follows approximately the same plan as that for UUT end-to-end test software. However, the plan is cycled only once for the end-to-end requirements; but for diagnostics, the plan is repeated a number of times (depending upon the complexity of the UUT). Diagnostic requirements are specified for each set of faults. Also, considerable circuit analysis must be accomplished to determine and understand what failures could cause similar or identical UUT malfunctions and to generate unique diagnostic test flow diagrams.

4.7.4 ATE Station Test Software Requirements Development

ATE station test software requiremants are concerned with ATE station self-test as defined in paragraph 4.1.3 under test station test software. The requirements are developed in precisely the same manner as for UUT test software requirements development (paragraph 4.7.3). Source data is defined in paragraph 4.7.1.3.

4.7.5 ITA Test Software Requirements Development

ITA test software requirements are concerned with ITA self-test. The requirements are developed by the same process as UUT test software requirements development (paragraph 4.7.3). Source data is defined in paragraph 4.7.1.2.

4.7.6 Test Software Development Specification

Test software requirements fall into two categories. These are UUT-dependent requirements; e.g., UUT test sequences, stimulus and measurement requirements; and ATE-dependent requirements relating to the ATE environment. Once the ATE has been selected, software requirements (principally functional and physical interfaces) are imposed on the UUT CPCIs. A development specification must contain the ATE-dependent requirements, UUT dependent test sequences and quality assurance provisions. The ATE-dependent and UUT-dependent requirements are discussed separately in the following two paragraphs.

4.7.6.1 ATE-Environment Requirements. These requirements are unique to a CPCI, even excluding the detailed UUT test sequences, stimulus and measurement requirements. ATE environment require-

ments definition can begin and a preliminary CPCI development specification prepared when:

a. ATE selection and procurement has progressed so that functional and physical interface requirements and constraints on UUT CPCIs can be established; and

b. UUT designs have progressed so that the requirements imposed on the performance, design, test, and qualification of the UUT test software can be established.

The details of the UUT performance and diagnostic tests themselves will not have been completed, but the functional areas will have been defined. For example, it will be possible to make statement such as:

> "This Test Software (CPCI) shall be subdivided into the following three functional areas:
>
> a. Mode Control
> b. Protection Circuit Tests
> c. Requlator Output Tests."

The CPCI Development Specification may be preliminary, and should contain:

UUT Test Computer Program (CPCI) Definition
- Interface requirements and description
  - Physical interfaces UUT to ATE/ITA
  - Functional interfaces, e.g., Electrical
  - Control Software (if required)
- Functional Requirements and Description
  - UUT Test identified and described functionally, e.g.,
  - Inputs
  - Processing
  - Outputs

Quality Assurance Requirements
- Type and extent of verification required
- Inspection
- Analysis
- Test
- Demonstration
- Test Environment
- Test Requirements, e.g. Correlation of type of test to section 3 requirements

As can be seen, a general understanding of the UUT functions and specific understanding of ATE capabilities is required to prepare the development specification as described above. This allows effective management of not only the requirements specification process, but also all of the subsequent development.

4.7.6.2 UUT Dependent Requirements. Test software CPCI development specifications must, as a minimum, identify the UUT tests to be performed and reference the appropriate TRD data that will contain the detailed test sequences, stimulus and responses. Since the TRD data, particularly diagnostic tests, is usually produced well downstream from the CPCI development specification these data are not available when the development specification is generated. The ATE environment requirements are available early and with the identification of the test sequences and appropriate references to TRD data, should be sufficient for the development specification.

If the TRD data are to be written directly in ATLAS, the development specification should include specification requirements for writing the ATLAS statements for the purpose of ensuring the resulting source code is compatible with the rest of the test software CPCI.

4.8 ATE SOFTWARE REQUIREMENTS SUMMARY

The ATE requirements specification process begins with a ROC for a weapon system. The reference here may only state that adequate support equipment be provided for effective maintenance. The weapon systems RFP may contain significant ATE requirements particularly if ATE development is to be included in the prime contract. However, the real beginning is the definition of the ATE resulting from a LSA and documented in a SERD. Following approval of the SERDs and the appropriate contract arrangements, work may begin specifically on ATE software requirements specification. ATE software falls into three general classifications: Control software, support software and test software. Test software is further divided into UUT test software, station test software and ITA test software. Control software, support software and station test software are all dependent on the ATE and its environment. The schedule for requirements specifications for these types of software is similar and occurs prior to the test software. The remaining test software; i.e., UUT and ITA test software, are dependent on production UUT's and the requirements specification process must of necessity lag behind the other types. Given the various sources of ATE requirements to specification process is not unlike any other software requirements specification process, requiring in the final reckoning, a development specification stating the required interface, functions to be performed, performance required and quality assurance provisions.

The ATE software requirements specification process is largely a contractor activity that must be monitored closely by the Air Force ATE software manager/engineer. The Air Force has the opportunity to influence the process in the preparation of the weapon system RFP, guidance and consulting during the LSA process, approval of the SERDs, preparation of the SOW for the contract supplement agreement, guidance and consulting in the preparation of the ATE Prime item development specification for ATE procurement (possibly approval, if so stated in the contract supplement) and in participating in the schedule PDRs and CDR for the various CPCIs.

Table 4.8-1 provides a checklist of significant considerations that should be made in the requirements specification process.

Table 4.8-1. ATE Software Requirements Specification Checklist

1. Has the contractual method for acquiring ATE been decided and identified: i.e., inclusion in the weapon system contract, contract supplement, or separate contract?

2. Does the weapon system contract contain provisions for a logistics support analysis and does the CDRL require SERD's for ATE?

3. Does the LSA specifically address ATE?

4. Are there SERD's that define the ATE?

5. Does the SOW for a contract supplement for ATE procurement contain definitive words regarding the tasks to be performed and the quantity and quality of the expected products?

6. Does the SOW/CDRL for the contract supplement require development and product specifications for each ATE CPCI? Does it contain provisions for all necessary data rights?

7. Has sufficient lead time been scheduled for the development of ATE software requirements with respect to the availability of the production UUT?

8. Has a prime item development specification been prepared for ATE suppliers and does it include requirements for ATE control and support software?

9. Does the SOW for the ATE supplier require a development specification for all newly developed software and a product specification or equivalent data for all delivered software? Are there provisions for technical reviews such as PDR's and CDR's?

10. Does the SOW for the ATE supplier provide for all necessary data rights?

11. Will the TRD data specifying UUT test sequences be written in English or ATLAS?

12. Have all PDR's and CDR's been attended?

13. Do the requirements for control software contain the provisions specified in Figure 4.1-1?

14. Do the requirements for support software contain the provisions specified in Figure 4.1-2?

15. Do the requirements for test software contain the provisions specified in Figure 4.1-3?

16. Do the test software CPCI development tests contain appropriate data as specified in paragraph 4.7.6.1?

17. Have total workload requirements for the ATE been defined? How many UUTs must be tested simultaneously? Has the worst cast situations been identified?

Section 5.0 BIBLIOGRAPHY

The following documents are applicable to the subject of requirements specification for ATE and TS software:

1. DODD 5000.29, Management of Computer Resources in Major Defense System, 26 April 1976

2. AD/A-004 547, Proceedings of the Aeronautical Systems Software Workshop, John H. Manley et al, 1 July 1974.

3. AFR 800 14 Vol II, Acquisition and Support Procedures for Computer Resources in Systems, 26 September 1975.

4. Safeguard Data-Processing System: Software Data Management, J. D. Musa and F. N. Woomer, Jr., from the Bell System Technical Journal, Safeguard Supplement, 3 January 1975.

5. MIL STD 483, Configuration Management Practices for Systems, Equipment Munitions, and Computer Programs, 1 June 1971.

6. MIL-STD-490, Military Standard Specification Practices, 18 May 1972.

7. DOD 5000.19L Vol. II, Acquisition Management Systems and Data Requirements Control List, January 1977. (DID descriptions)

8. MIL-STD-1519, Preparation of Test Requirements Documentation, 17 September 1971.

9. AFR 800-8, ILS Program for Systems and Equipment, 27 July 1972.

10. MIL-D-83468, Digital Computational System for Real-Time Training Simulators, 12 December 1975.

11. AFLC Regulation 66-37, Management of Automated Test Systems, 24 October 1975.

12. MIL-S-83490, Specifications, Types and Forms, 30 October 1968.

13. MIL-STD-499A, Engineering Management, 1 May 1974.

14. AFR 57-1, Required Operational Capabilities, 30 May 1975.

Section 6.0 MATRIX: GUIDEBOOK TOPICS VS. GOVERNMENT DOCUMENTS

The elements in Figure 6.0-1 correspond to the sections in the government publication wherein the corresponding topic is discussed to the largest extent.

TOPICS \ GOVERNMENT PUBLICATION	MIL-D-83468	AFLCR 66-37	AFR 800-14 VOL II	MIL-STD-483	MIL-STD-490	MIL-STD-83490	MIL-STD-499A	DODD 5000.29	MIL-STD-1519	AFR 57.1
TRAINER SIMULATOR	1.0									
AUTOMATIC TEST EQUIPMENT		1.0								
SYSTEM ENGINEERING			4.4 4.5 4.7	3.3.1						
ROC			3.2							●
DEVELOPMENT SPECIFICATION			2.5 3.3	3.4.7 3.4.8 APP.VI	3.1.3.2 3.1.3.2.5 APP.VI	3.2.2				
SUPPORT EQUIPMENT	3.3									
SYSTEM SPECIFICATION				APP.III	3.1.3.1 APP.I	3.2.1				

Figure 6.0-1. Guidebook Topics Versus Government Documentation (Sheet 1 of 2)

TOPICS \ GOVERNMENT PUBLICATION	MIL-D-83468	AFLCR 66-37	AFR 800-14 VOL II	MIL-STD-483	MIL-STD-490	MIL-STD-83490	MIL-STD-499A	DODD 5000.29	MIL-STD-1519	AFR 57.1
FUNCTIONAL ANALYSIS			2.8				10.2.2			
REQUIREMENTS ALLOCATION							10.2.3			
ATE SOFTWARE		1.0								
CPCI				3.4.7 APP.VI				VC		
TRD		1.0							●	
TS COMPUTATIONAL SYSTEM	3.0 3.4 4.3.1									
TS SOFTWARE	3.2									

Figure 6.0-1. Guidebook Topics Versus Government Documentation (Sheet 2 of 2)

Section 7.0 GLOSSARY OF TERMS

Acquisition Engineer - Military or civilian member of a SPO or an AFSC division who supports the activities of a SPO.

Computer Program - A series of instructions or statements in a form acceptable to computer equipment, designed to cause the execution of an operation or series of operations. Computer programs, and maintenance/diagnostic programs. They also include applications programs such as payroll, inventory, control, operational flight, strategic, tactical, automatic test, crew simulator and engineering analysis programs. Computer programs may be either machine dependent or machine independent, and may be general purpose in nature or be designed to satisfy the requirements of a specialized process of a particular user.

Computer Program Comfiguration Items - A computer program or aggregate of related computer programs designated for configuration management. A CPCI may be a punched deck of cards, paper or magnetic tape or other media containing a sequence of instructions and data in a form suitable for insertion in a digital computer.

Configuration Item - An aggregation which satisfies an end use function and is designated for configuration management.

Configuration Control - A management discipline applying technical and administrative direction and surveillance to:

a. Identify and document the functional and physical characteristics of a configuration item

b. Control changes to those characteristics; and

c. Record and report change processing and implementation status

Control Software - Software used during execution of a test program which controls the nontesting operations of the ATE. This software is used to execute a test procedure but does not contain any of the stimuli or measurement parameters used in testing a unit under test. Where test software and control software are combined in one inseparable program, that program will be treated as test software (AFLC 66-37).

Data Base - A collection of program code, tables, constants, interface elements and other data essential to the operation of a computer program or software subsystem.

External Interface - Data passed between two or more computer programs or between a computer program and peripheral devices external to the computer in which the program resides. The data may be in the form of an interrupt signal or may be a digital data stream either output from the computer or input into the computer for processing.

Instructional System - That portion of a TS which supports the instructor's functions. It consists of hardware and software used by the instructor to communicate with trainees to control the state of the simulator by insertion of faults and to display simulator status and student responses.

Internal Interfaces - Data passed between elements of a computer program and usually included in the computer program data base.

Logic Flow - A diagrammatic representation of the logic sequence for a computer program. Logic flows may take the form of the traditional flow charts or in some other form such as a program design language.

Organic - A term used to designate a task performed by the Air Force rather than a contractor.

Product Baseline - The final approved configuration identification. It identifies the as designed and functionally tested computer program configuration. It is defined by the Computer Program Product Specification.

Program Design Language - An English-like, specially formatted, textual language describing the control structure, logic structure, and general organization of a computer program. Essential features of a program design language are:

a. It is an English-like representation of a computer procedure that is easy to read and comprehend.

b. It is structured in the sense that it utilizes the structured programming control structures and indentation to show nested logic.

c. It uses full words or phrases rather than the graphic symbols used in flow charts and decision tables.

Quality Assurance - A planned and systematic pattern of all software-related actions necessary to provide adequate confidence that computer program configuration items or products conform to establish software technical requirements and that they achieve satisfactory performance.

Software - A combination of associated computer programs and computer data required to enable the computer equipment to perform computational or control functions.

Software Quality - The primary characteristic of software quality is that the software performs as intended. This implies not only that the software reflects the specification to which it is written but also that the software specifications themselves adequately address the system/mission requirements. Key attributes of software quality include: reliability, flexibility, traceability, testability, integrity, maintainability, and completeness. Quality software is: well-defined, well-documented, free of design deficiencies and coding errors, satisfies performance requirements, and has minimum life cycle cost.

Source Selection - The process of selecting which among competing contractors shall be awarded a contract. A significant portion of this involves evaluation of proposals to determine the degree to which the government's requirements would be satisfied.

SPO Cadre - Nucleus of a SPO formed by an AFSC division in accordance with AFR 800-2.

Support Software - Auxiliary software used to aid in preparing, analyzing and maintaining other software. Support software is never used during the execution of a test program on a tester, although it may be resident either on-line or off-line. Included are assemblies, compilers, translators, loaders, design aids, test aids, etc. (AFLC 66-37).

System Engineering - The application of scientific and engineering efforts to transform an operational need or statement of deficiency into a description of systems requirements and a preferred system configuration that has been optimized from a life cycle viewpoint. The process has three principal elements: functional analysis, synthesis, and trade studies or cost-effectiveness optimization.

Test Software - Programs which implement documented test requirements. There is a separate test program written for each distinct configuration of unit under test (AFLC 66-37).

Top Down Structured Programs - Structured programs with the additional characteristics of the source code being logically, but not necessarily physically, segmented in a hierarchical manner and only dependent on code already

written. Control of execution between segments is restricted to transfer between vertically adjacent hierarchical segments.

Validation - Computer program validation is the test and evaluation of the complete computer program aimed at ensuring compliance with the performance and design criteria.

Verification - Computer program verification is the iterative process of continuously determining whether the product of each step of the computer program acquisition process fulfills all requirements levied by the previous step, including those set for quality.

System Life Cycle - The system acquisition life cycle consists of the following five major phases with major decision points:

a. Conceptual phase

b. Validation phase

c. Full-scale development phase

d. Production phase

e. Deployment phase

(AFR-800-14, Volume II)

Section 8.0 ABBREVIATIONS AND ACRONYMS

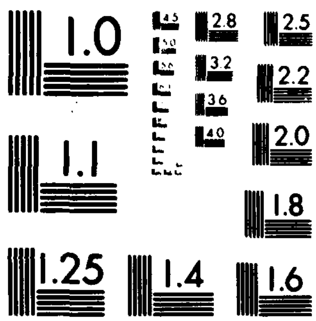
ADI	Attitude Direction Indicator
AF	Air Force
AFLC	Air Force Logistics Command
AFSC	Air Force Systems Command
AGERD	Aerospace Ground Equipment Requirements Documentation
AI	Adapter Interface
ASD	Aeronautical Systems Division
ATE	Automatic Test Equipment
ATLAS	Abbreviated Test Language for All Systems
ATPG	Automatic Test Pattern Generator
CCP	Contract Change Proposal
CDR	Critical Design Review
CDRL	Contract Data Requirements List
CEI	Contract End Item
CPCI	Computer Program Configuration Item
CPDP	Computer Program Development Plan
CPU	Central Processing Unit
CRISP	Computer Resources Integrated Support Plan
CRT	Cathode Ray Tube
CRWG	Computer Resources Working Group
DCP	Development Concept Paper
DDP	Design Data Package
DID	Data Item Description
DRLMS	Digital Radar Land Mass Simulator
FEMA	Failure Modes and Effects Analysis
FORTRAN	Formula Translator
HOL	High Order Language
HSI	Horizontal Situation Indicator
I/O	Input/Output
IOC	Initial Operational Capability
ISD	Instructional Systems Development
ITA	Interface Test Adapter
LCC	Life Cycle Cost
LRU	Line Replaceable Unit
LSA	Logistics Support Analysis
LSC	Logistics Support Costs
MTBO	Mean Time Between Overhaul
NSCCA	Nuclear Safety Cross-Check Analysis
ORLA	Optimum Repair Level Analysis
PD	Preliminary Design
PDR	Preliminary Design Review
PMD	Program Management Directive
PMP	Program Management Plan
RDT&E	Research Design Test and Evaluation
RFI	Radio Frequency Interference

RFP	Request for Proposal
ROC	Required Operational Capabilities
SAE	Software Acquisition Engineering
SERD	Support Equipment Recommendations Data
SOW	Statement of Work
SPO	System Program Office
SRU	Secondary Replacement Unit
TI	Technical Interchange
T.O.	Technical Order
TRD	Test Requirement Document
TS	Trainer Simulator
UUT	Unit Under Test
V&V	Verification and Validation

AD-A083 207   BOEING AEROSPACE CO SEATTLE WA   F/G 9/2
REQUIREMENTS SPECIFICATION. ONE OF THE SOFTWARE ACQUISITION ENG--ETC(U)
JAN 79 D C WHITMORE   F33657-76-C-0723
UNCLASSIFIED D180-20676-1   ASD-TR-78-45   NL

2 · 2

END
DATE
FILMED
5 80
DTIC


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5

Section 9.0 REQUIREMENTS SPECIFICATION INDEX

9.1 Automatic Test Equipment

9.1 Automatic Test Equipment (Continued)

9.2 Trainer Simulator

TOPIC	PARAGRAPH
Analyses & Trades (TS)	3.1.2, 3.1.2.2, 3.1.5, 3.1.6
Alternative System Evaluation	3.1.4, 3.1.4.1, 3.1.4.2
Computer Program Development Plan	3.2.1
Computer Resources	3.2.1, 3.2.1.1, 3.2.1.2, 3.2.1.3
Computer Resources Integrated Support Plan	3.2.1, 3.2.1.1, 3.2.1.2
Contractor Technical Proposal	3.1.5, 3.1.6, 3.2.1.3, 3.3.1, 3.3.2
Design Data Package	3.3.2
Documentation (TS)	3.0, 3.1.3, 3.2, 3.2.1.1, 3.3
Hardware (TS)	1.3.1, 3.0, 3.1.2.1
Hardware/Software Trade-offs (TS)	3.0, 3.1.5
Instructional Systems Development	3.1.1
MIL-D-83468	3.0, 3.3, 3.3.1, 3.3.2
Non-real Time Software (TS)	3.1.2.1, 3.1.3, 3.1.5
Planning (TS)	3.0, 3.2
Preliminary Design (TS)	3.1.2, 3.1.3
Problem Areas	3.4
Program Management Plan	3.2.1, 3.2.1.1, 3.2.1.2
Real Time Software (TS)	3.1.2.1, 3.1.3, 3.1.5
Requirement Change (TS)	3.4
TS Requirements Specification	1.2, 3.0
ROC (TS)	1.2, 3.0, 3.1, 3.1.1, 3.1.2, 3.1.2.2, 3.1.3, 3.3.2
RFP	3.0, 3.1.5, 3.3.1, 3.3.2
Software Engineer (TS)	3.0, 3.3

9.2 Trainer Simulator (Continued)

TOPIC	PARAGRAPH
Software Preliminary Design (TS)	3.1.5
SPO	3.1.1, 3.1.3, 3.1.4, 3.3.2
System Characteristic	1.3.1
System Preliminary Design Checklist (TS)	3.1.3
System Specification (TS)	3.0, 3.1.4, 3.1.5, 3.3, 3.3.1, 3.3.2
Technical Evaluation (TS)	3.0, 3.1
Training Instructor Control	3.1.2.2, 3.1.3, 3.1.5, 3.1.6
Training Simulation	3.1.1
Training System Requirements	3.1.2, 3.1.2.1, 3.1.2.2

*U.S.Government Printing Office: 1980 — 657-084/529